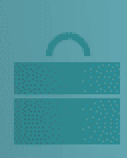


一本讲透基础知识、实例开发、模块开发、项目开发的百科全书



学习测试、诊断
网站提供编程能力测试、软件考试模拟测试题库。（登录网站）

有趣实践任务
光盘提供580多个实践任务，读者可以登录网站获取答案。（光盘+网站）



赠送开发案例
赠送开发案例文档、源程序和学习视频，帮助读者拓展视野，提高熟练度。（光盘中）



学习经验分享
提供互动、互助学习平台，学习分享经验。（登录网站）

专业资源库
免费赠送程序开发资源库（学习版），拓展编程视野。（登录网站）

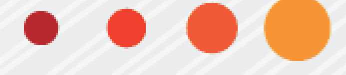
39小时专业视频讲解
389个实例、模块、项目分析

软件开发技术联盟 编著

PHP+MySQL开发实战

带着任务去学习，在编程环境中学编程

39小时专业学习视频、389个实战范例、强大学习资源包（学习测试诊断、有趣实践任务、专业资源库、在线交流、学习经验分享、项目案例分享、习题与解答、源程序等）



清華大学出版社

软件开发实战

PHP+MySQL 开发实战

软件开发技术联盟　编著

清華大学出版社
北京

内 容 简 介

《PHP+MySQL 开发实战》从初学者的角度讲述使用 PHP 语言结合 MySQL 数据库进行程序开发应该掌握的各项技术，内容突出“基础”、“全面”、“深入”的特点，同时强调“实战”效果。书中在介绍技术的同时提供实例，同时在各章的结尾安排有实战，通过实战来综合应用本章所讲解的知识，做到理论联系实际；每篇的最后一章有一个综合实例，通过一个模块综合讲解本篇所讲解的知识内容；在本书的最后两章中提供了两个完整的项目实例，讲述从前期规划、设计流程到项目最终实施的整个实现过程。

全书共分 28 章，主要内容包括初识 PHP 环境搭建、PHP 语言基础、PHP 流程控制语句、字符串操作与正则表达式、初探数组、日期和时间的管理、程序调试与异常处理、综合实例（一）——在线论坛、MySQL 数据库、MySQL 存储引擎与运算符、MySQL 函数之选、MySQL 基本操作、MySQL 数据查询、综合实例（二）——留言本、MySQL 存储过程和函数、MySQL 事务、触发器、综合实例（三）——物流管理系统、ADODB 类库、数据库编程技术、PDO 数据库抽象层、综合实例（四）——BCTY365 网上社区、Smarty 模板技术、ThinkPHP 框架、Zend Framework 框架、综合实例（五）——电子商务网站、易查供求信息网和图书馆管理系统。所有知识都结合具体实例进行介绍，对涉及的程序代码给出了详细的注释，读者可以轻松领会 PHP+MySQL 程序开发的精髓，快速提高开发技能。本书特色及丰富的学习资源包如下：

黄金学习搭配、专业学习视频、重难点精确打击、学习经验分享、学习测试诊断、有趣实践任务、专业资源库、学习排忧解难、获取源程序、提供习题答案、赠送开发案例。

本书适合有志于从事软件开发的初学者、高校计算机相关专业学生和毕业生，也可作为软件开发人员的参考手册，或者高校的教学参考书。

图书在版编目（CIP）数据

PHP+MySQL 开发实战/软件开发技术联盟编著．—北京：清华大学出版社，2013
（软件开发实战）

ISBN 978-7-302-31895-8

I. ①P…　II. ①开…　III. ①PHP 语言–程序设计　②关系数据库系统　IV. ①TP312　②TP311.138

中国版本图书馆 CIP 数据核字（2013）第 074845 号

责任编辑： 赵洛育
封面设计： 陈　敏
版式设计： 文森时代
责任校对： 张彩凤
责任印制：
出版发行： 清华大学出版社
　　网　　址： http://www.tup.com.cn，http://www.wqbook.com
　　地　　址： 北京清华大学学研大厦 A 座　　**邮　　编：** 100084
　　社 总 机： 010-62770175　　**邮　　购：** 010-62786544
　　投稿与读者服务： 010-62776969，c-service@tup.tsinghua.edu.cn
　　质量反馈： 010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销： 全国新华书店
开　　本： 203mm×260mm　**印　　张：** 43.75　**字　　数：** 1449 千字
　　（附视频光盘、海量学习资源 DVD 1 张）
版　　次： 2013 年 9 月第 1 版　**印　　次：** 2013 年 9 月第 1 次印刷
印　　数： 1～4000
定　　价： 89.80 元

产品编号：052535-01

本书编著者名单

主　编：张　鑫　王雨竹

编　著：张　鑫　王雨竹　辛洪郁　王小科　王国辉　杨　丽
顾彦玲　赛奎春　高春艳　陈　英　宋禹蒙　刘　佳
刘莉莉　陈丹丹　隋光宇　郭　鑫　刘志铭　李　伟
张金辉　李　慧　刘　欣　李继业　潘凯华　赵永发
寇长梅　赵会东　王敬洁　李浩然　苗春义　刘清怀
张世辉　张　领

前　　言

Preface

PHP 和 MySQL 是全球最普及、应用最广泛的互联网开发语言及最流行的数据库之一。PHP 语言具有简单、易学、源码开放，可操纵多种主流与非主流的数据库，支持面向对象的编程，支持多种开源框架（如 Zend Framework），支持跨平台的操作，而且完全免费等特点，而 MySQL 越来越受到广大程序员的青睐和认同。全球最大的网络搜索引擎公司 Google 使用的数据库就是 MySQL，并且国内很多大型网络公司也选择 MySQL 数据库，而 PHP 目前拥有几百万个用户，其发展速度要快于在它之前的任何一种计算机语言。相信 PHP 和 MySQL 一定能够经得起实践的检验，发展成为互联网开发语言中“主流的主流”。

本书特色及配套学习资源包

为了方便读者学习，本书经过了科学安排，并配备了丰富的学习资源包，读者朋友可从本书的配书光盘或者网站 www.rjkflm.com 获取学习资源。

黄金学习搭配 快速入门+中小实例实战+模块实战+项目实战+开发资源包。 （图书+光盘+网站）	**专业学习视频** 光盘含 39 小时大型同步教学视频，听专家现场演示讲解。 （光盘中）	**重难点精确打击** 305 个精彩实例分析，精确掌握重点难点。（图书）
学习分享经验 提供互动、互助学习平台，学习分享经验。（登录网站）	**学习测试、诊断** 网站提供编程能力测试、软件考试模拟测试题库。 （登录网站）	**有趣实践任务** 光盘提供 580 多个实践任务，读者可以登录网站获取答案。 （光盘+网站）
专业资源库 免费赠送 PHP 程序开发资源库（学习版），拓展编程视野。 （登录网站）	**学习排忧解难** 提供编程学习论坛，头脑风暴，帮您轻松解决编程困扰。 （登录网站）	**获取源程序** 光盘提供几乎所有的实例源程序，可直接复制，照猫画虎，调试运行。（光盘中）
提供习题答案 本书对于习题都给出了答案，先自行作业，然后对比分析。 （光盘中）	**赠送开发案例** 赠送开发案例文档、源程序和学习视频，帮助读者拓展视野，提高熟练度。（光盘中）	

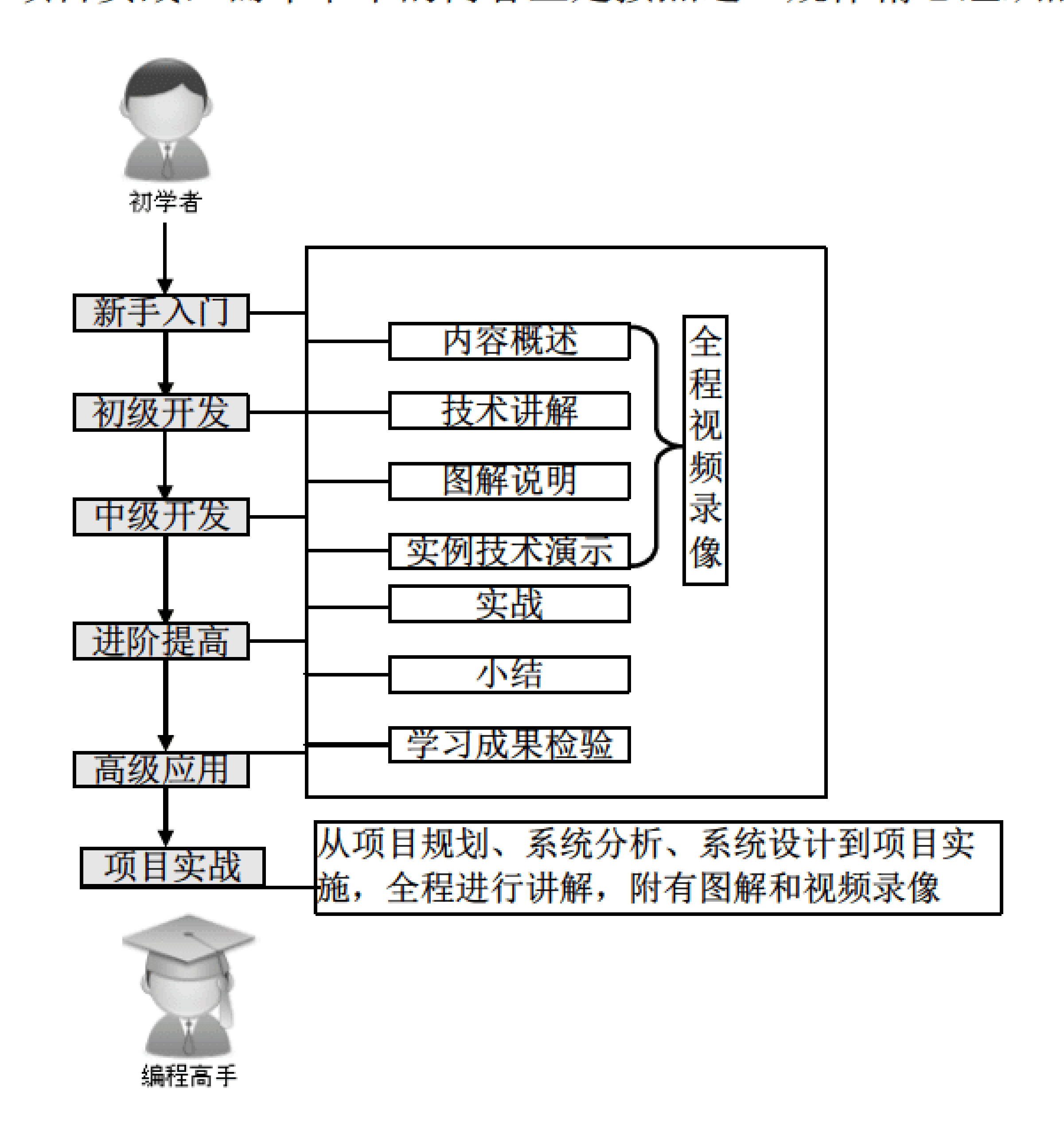
读者对象

- ☑ 有志于从事软件开发的初学者
- ☑ 高等院校计算机相关专业的老师和学生
- ☑ 准备从事软件开发的求职者
- ☑ 参与毕业设计的学生
- ☑ 初、中级程序开发人员
- ☑ 程序测试及维护人员

本书内容结构

从初学程序开发的人员步入编程高手行列通常需要经历 6 个阶段，即新手入门—初级开发—中级开发

—进阶提高—高级应用—项目实战，而本书中的内容正是按照这一规律精心组织的，结构如下图所示。



第 1 篇：新手入门。其主要内容包括初识 PHP 环境搭建、PHP 语言基础、PHP 流程控制语句、字符串操作与正则表达式、初探数组、日期和时间的管理、程序调试与异常处理、综合实例（一）—— 在线论坛。

第 2 篇：初级开发。其主要内容包括 MySQL 数据库、MySQL 存储引擎与运算符、MySQL 函数之选、MySQL 基本操作、MySQL 数据查询、综合实例（二）—— 留言本。

第 3 篇：中级开发。其主要内容包括 MySQL 存储过程和函数、MySQL 事务、触发器、综合实例（三）—— 物流管理系统。

第 4 篇：进阶提高。其主要内容包括 ADODB 类库、数据库编程技术、PDO 数据库抽象层、综合实例（四）—— BCTY365 网上社区。

第 5 篇：高级应用。其主要内容包括 Smarty 模板技术、ThinkPHP 框架、Zend Framework 框架、综合实例（五）—— 电子商务网站。

第 6 篇：项目实战。通过两个完整的项目实例介绍了大型应用程序的设计过程，包括易查供求信息网和图书馆管理系统。这两个项目是作者精心挑选的，内容覆盖多个方面。通过对这两个项目的学习，读者可以巩固前面所学的知识和技术，积累项目开发经验。

本书备用服务

如果本书服务网站 www.rjkflm.com 临时有问题，读者朋友还可以通过如下方式与我们沟通：登录网站：www.mingribook.com，查阅相关问题或者留言。通过 QQ：4006751066。

本图书光盘如有打不开现象，请核实一下电脑是不是 DVD 光驱；如果在复制光盘内容时，出现个别文件无法复制，请分批复制试一试；如有极个别光盘打不开，可多试几台电脑，打开之后复制内容一样使用。

“宝剑锋从磨砺出，梅花香自苦寒来”，亲爱的读者朋友，希望在辛苦的道路上我们一起走过！

编　者



目　　录

Contents

第 1 篇　新手入门

第 2 篇 初级开发



第 12 章 MySQL 基本操作 232

视频讲解：37 分钟

第 13 章 MySQL 数据查询 247

视频讲解：60 分钟



第 3 篇 中级开发

第 4 篇　进阶提高



第 5 篇 高级应用



第 6 篇　项目实战



第 1 篇

新手入门

第 1 章

初识 PHP 环境搭建

（视频讲解：66 分钟）

要使用 PHP，首先要建立 PHP 开发环境。PHP 是全球最普及、应用最广泛的互联网开发语言之一。学习任何一门编程语言，在开始学习前都要首先学会搭建和熟悉开发环境。本章将介绍两种操作系统（Windows 和 Linux）下的 Apache 服务器、MySQL 服务器及 PHP 的安装和配置方法。另外，还特别为初学者介绍一种简化安装和配置 PHP 环境的组合包。最后，使用 Dreamweaver 开发 PHP 的第一个实例。

通过阅读本章内容，你可以：

- **掌握如何在 Windows 和 Linux 下安装和配置 Apache 服务器**
- **掌握如何在 Windows 和 Linux 下安装 PHP**
- **掌握如何使用组合包快速搭建 PHP 环境**
- **掌握如何应用开发工具编写、发布和运行第一个 PHP 程序**
- **掌握如何独立解决常见的 PHP 环境安装问题**

1.1　PHP 开发环境的搭建

1.1.1　在 Windows 下搭建 PHP 开发环境

在 Windows 下搭建 PHP 与安装其他的一些软件工具不同。因为 PHP 是从 Linux 移植过来的一种语言，不仅在开发环境上保留着 Linux 的特点（Apache 是 Linux 下的 Web 服务器，地位就像 Windows 下的 IIS；MySQL 也是 Linux 系统中捆绑的数据库），在安装上也被烙上了 Linux 印记。除了正常的安装操作外，还需要在各自的配置文件（.ini、.conf）中进行专门的设置。

安装之前要准备的安装包有：

☑ Apache_2.2.8-win32-x86-no_ssl.msi。下载地址为 http://httpd.Apache.org/download.cgi。
☑ php-5.2.5-Win32.zip。下载地址为 http://www.php.net/downloads.php。
☑ mysql-essential-5.0.51a-win32.msi。下载地址为 http://www.mysql.com/download/（下载 MySQL 需要注册一个账号）。

1.1.2　在 Linux 下搭建 PHP 开发环境

在 Linux 下搭建 PHP 环境比 Windows 下要复杂得多。除了 Apache、PHP 等软件外，还要安装一些相关工具，设置必要的参数。而且，要使用 PHP 扩展库，还要进行编译。如本书中使用到的 SOAP、MHASH 等扩展库。这里给出在 Linux 下安装的必要步骤。如果用户在安装过程中遇到特殊的问题，还需要翻阅 Linux 相关的书籍和手册。

安装之前要准备的安装包有：

☑ httpd-2.2.11.tar.gz。下载地址为 http://www.apache.org。
☑ php-5.2.5.tar.gz。下载地址为 http://www.php.net/downloads.php。
☑ mysql-5.0.51a-Linux-i686.tar.gz。下载地址为 http://www.mysql.com。
☑ libxml2-2.6.26.tar.gz。可在网络上直接搜索该版本进行下载。

1.2　Apache 服务器的安装和配置

视频讲解：光盘\TM\Video\第 1 章\Apache 服务器的安装和配置.exe

1.2.1　在 Windows 下安装 Apache 服务器

Apache 服务器是全球范围内使用范围最广的 Web 服务软件，超过 50%的网站都在使用 Apache 服务器，它以高效、稳定、安全、免费（最重要的一点）的优势成为了最受欢迎的服务器软件。

本节主要介绍如何在 Windows 操作系统中安装和配置 Apache 服务器。安装 Apache 服务器前，应到官方网站 http://www.apache.org 下载 Apache 的安装程序 Apache_2.2.11-win32-x86-no_ssl.msi。

在 Windows 下安装和配置 Apache 服务器的操作步骤如下：

（1）双击 Apache_2.2.11-win32-x86-no_ssl.msi 文件，弹出欢迎页面。单击 Next 按钮，进入到许可协议

页面。

（2）在许可协议页面，用户需要同意页面中的条款才能继续安装。选中 I accept the terms in the license agreement 单选按钮，单击 Next 按钮进入到下一页面，如图 1.1 所示。

（3）图 1.1 所示页面是对该程序的一个描述和说明。在了解了相关的信息后，单击 Next 按钮进入到 Server Information 页面。

（4）Server Information 页面需要用户填写域名、服务器名称和管理员 Email，如图 1.2 所示。该页面还有两个单选按钮，如果选中默认的第一个单选按钮，说明该服务器对所有人开放，并且服务器的端口号为 80，这是推荐选项。选中第二个单选按钮是指该服务器仅对当前用户开放，并且服务器端口为 8080。这里选中第一个单选按钮，然后单击 Next 按钮进入下一个页面。

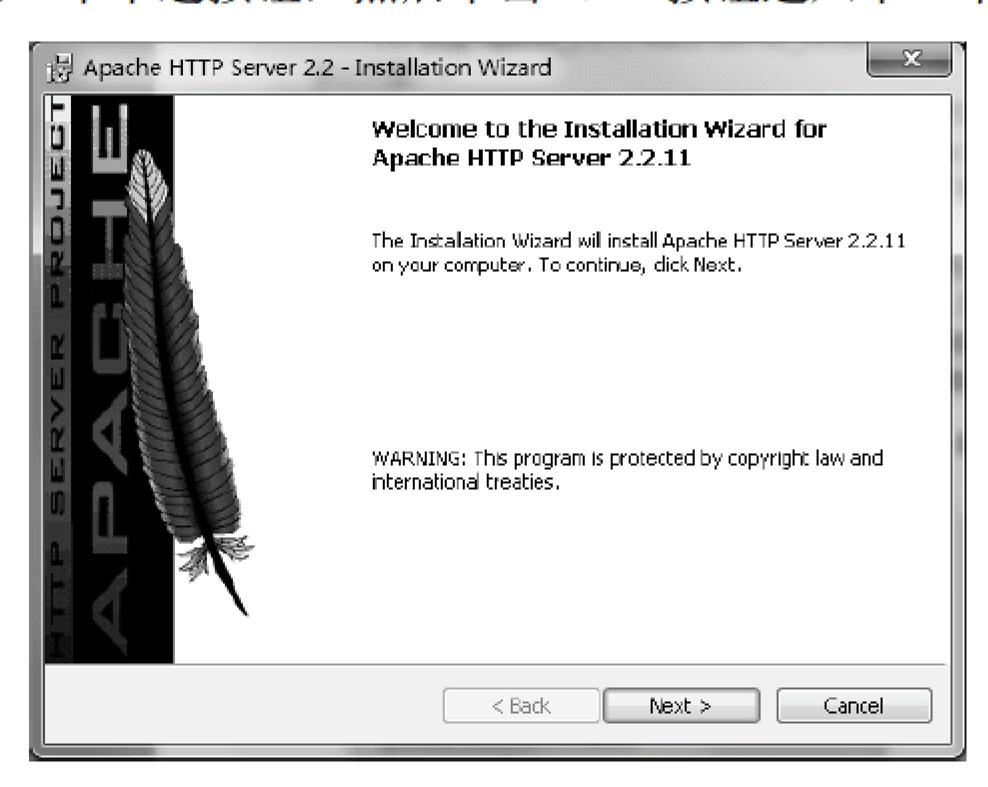


图 1.1　程序描述和说明页面

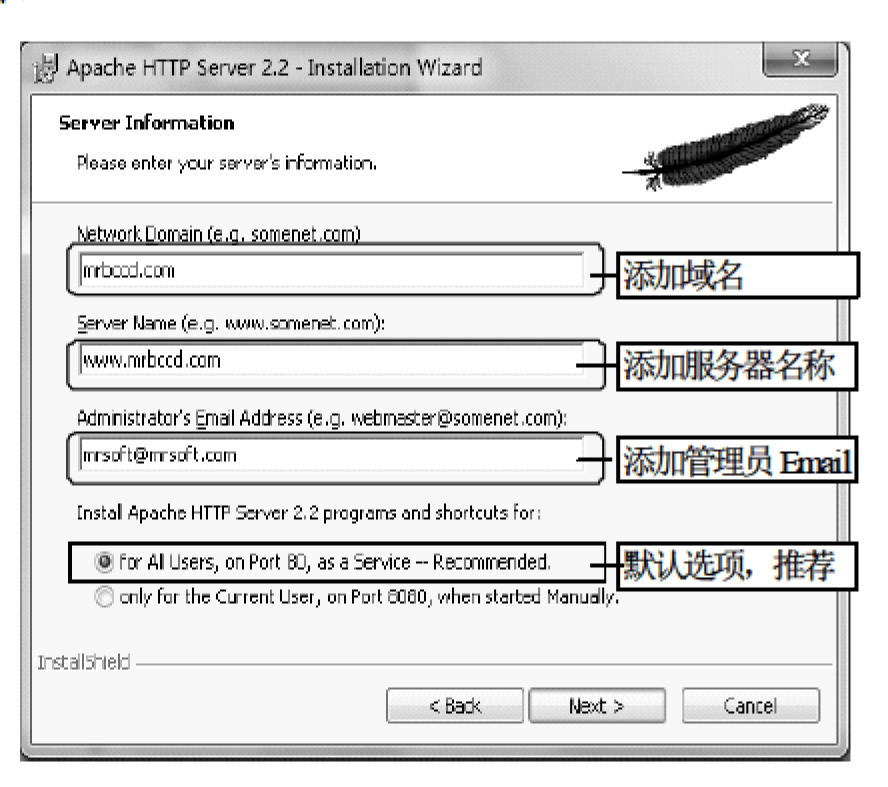


图 1.2　Server Information 页面

注意　如果用户的机器安装有 Internet 信息服务（IIS）管理器，那么必须将此项服务停止，因为 IIS 服务器的默认端口号为 80，同 Apache 服务器默认端口号相同。如果 IIS 服务不停止，就会和 Apache 服务器的端口号产生冲突，Apache 服务器将不能成功安装。

（5）如图 1.3 所示的页面用于选择安装类型。安装类型分为典型安装和自定义安装，通常保持默认设置即可。单击 Next 按钮，进入到路径选择页面。

（6）在路径选择页面中，单击 Change 按钮可以选择安装路径。这里路径设为 D:\Apache2.2\，如图 1.4 所示。

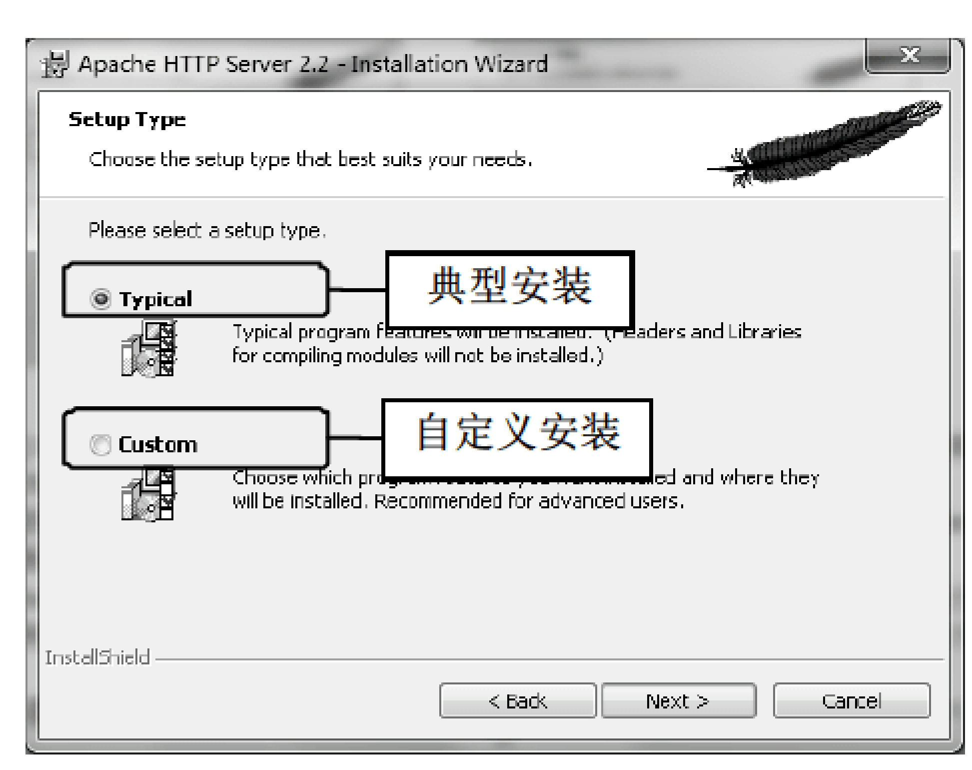


图 1.3　选择安装类型

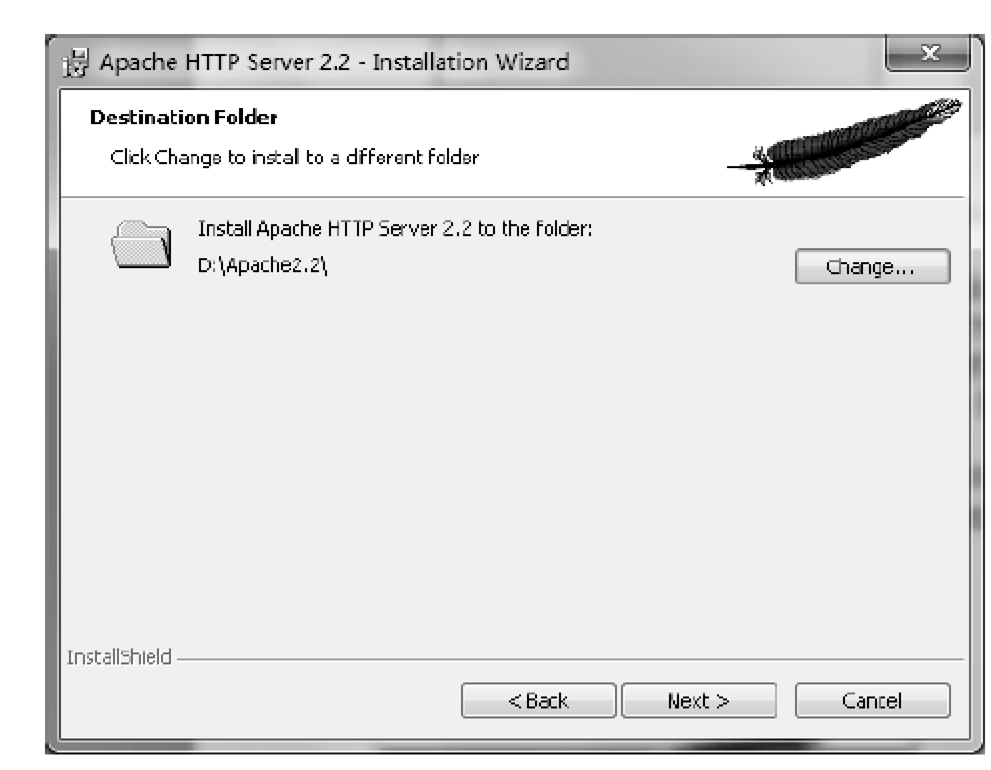


图 1.4　指定路径

（7）单击 Next 按钮进入文件安装页面。这是 Apache 安装的最后一步，程序开始安装文件。安装结束后，单击 Finish 按钮结束安装程序。

（8）安装完成后，Apache 服务器会自动开启。在系统托盘区域将出现一个图标，当前 Apache 服务启动时，图标样式为；服务器未启动时，图标样式为。

单击 Apache 服务器的启动小图标，将会看到服务器的开启与关闭功能；也可以右击小图标，在弹出的快捷菜单中选择 OpenApacheMonitor 命令，打开 Apache 监控程序，其操作效果如图 1.5 所示。

（9）服务器开启后，需要进行测试。打开 IE 浏览器页面，在地址栏中输入“http://127. 0.0.1/”或“http://localhost/”，按 Enter 键后系统会显示如图 1.6 所示的页面，此时说明 Apache 服务器安装成功。

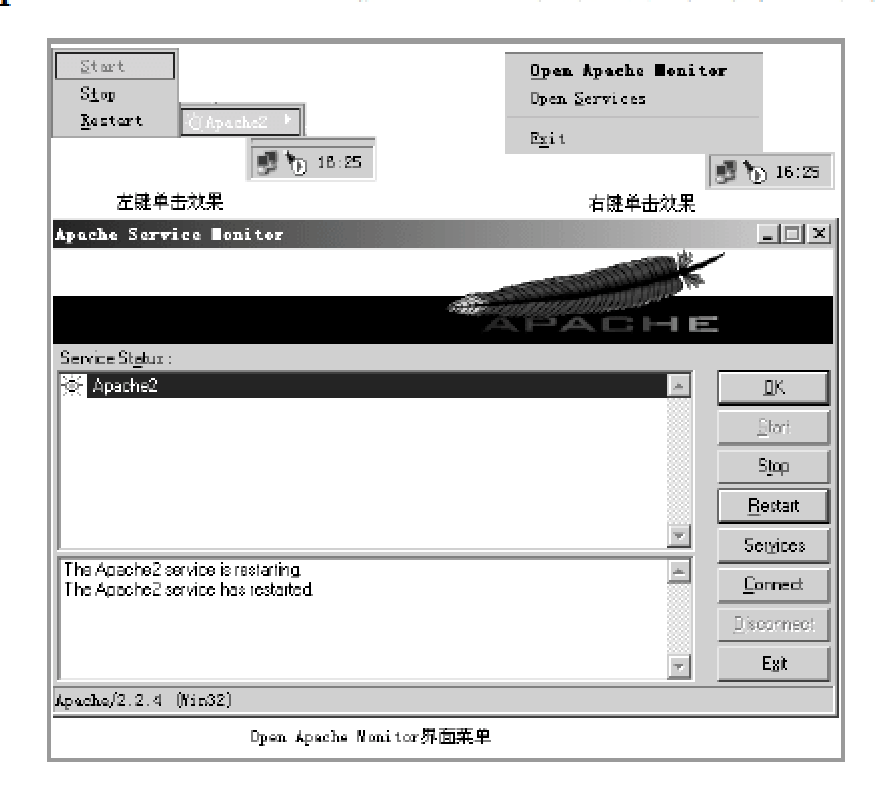


图 1.5　Apache 控制菜单

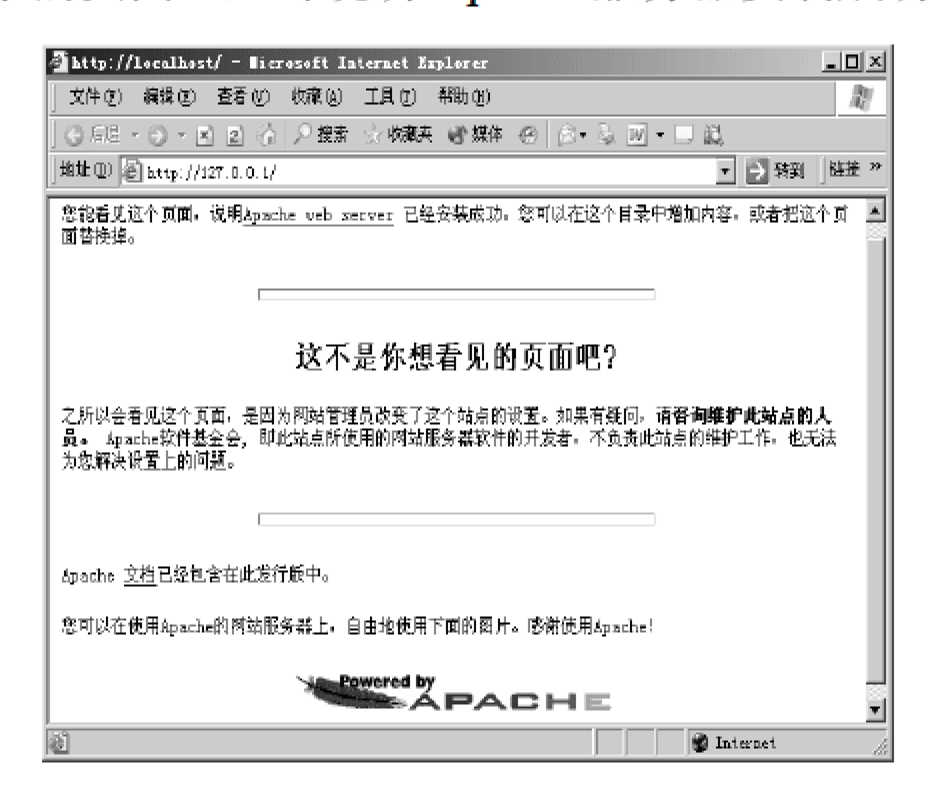


图 1.6　Apache 服务器运行页面

（10）Apache 服务器安装成功后，接下来需要对 Apache 服务器进行配置，以便 Apache 服务器能够识别 PHP 文件。配置 Apache 服务器主要是在 Apache 安装目录下的 conf 子目录的 httpd.conf 文件中进行，找到该文件并用记事本等文本编辑器打开该文件。

（11）定位到 LoadModule 配置块，在 LoadModule 的最后添加如下信息。

```
LoadModule php5_module d:\php5\php5Apache2_2.dll
```

添加后的文件效果如图 1.7 所示。



图 1.7　加载.dll 文件

（12）修改 DocumentRoot 参数可以修改 Apache 服务器主文档的根目录。原根目录的位置是 Apache2.2\htdocs，用户可以任意指定位置。如：

```
DocumentRoot "D:/Webpage"
```

在 DocumentRoot 的下面间隔约 28 行的位置，有一行“<Directory "D:/Apache2.2/htdocs">”，修改为“<Directory "D:/Webpage">”。



DocumentRoot 和这里的参数值要保持一致。

（13）添加 Apache 服务器能够识别的 PHP 扩展名。PHP 的扩展名有.php3、.php4、.php、.phtml 等。这里只推荐使用标准的扩展名.php。添加的代码如下：

```
AddType application/x-httpd-php .php
```

添加位置如图 1.8 所示。

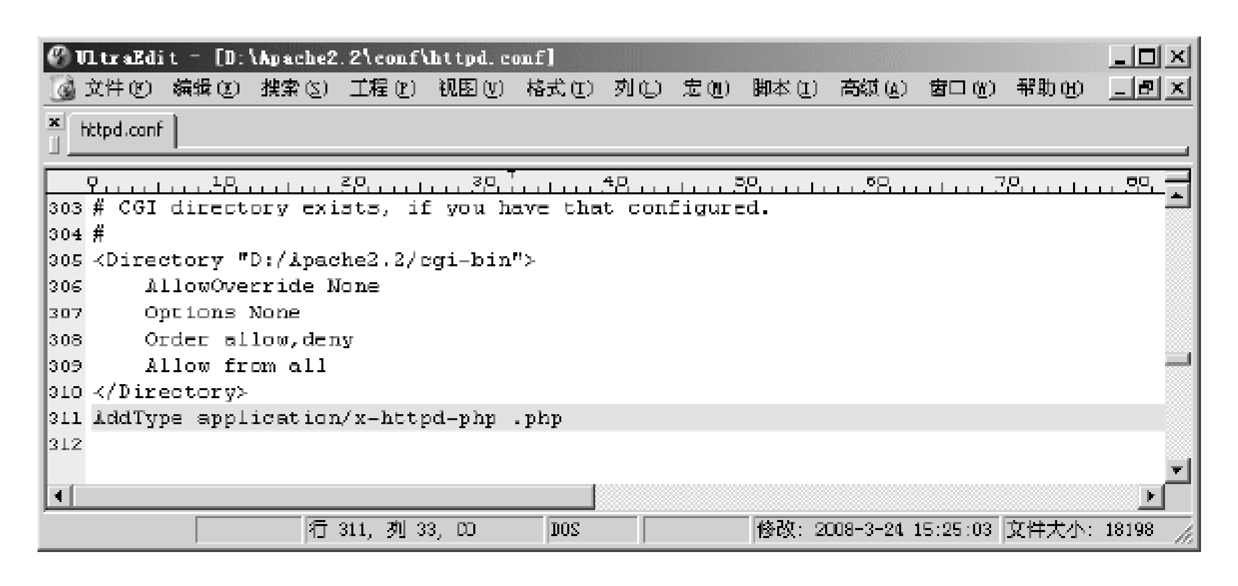


图 1.8　添加 PHP 扩展识别

（14）默认显示页。Apache 的默认显示页为 index.html。也就是说，在服务器未指名文件时，首先查找 index.html，如果找到 index.html，那么服务器就将加载该文件，否则显示目录内的文件列表。在这里添加一个 PHP 默认页：index.php。更改后的代码如下：

```
DirectoryIndex index.html index.php
```

（15）修改 Apache 端口号。Apache 的端口号为 80。修改 Listen 选项的值，即可修改端口号。如改为 82，则更改后的代码如下：

```
Listen 82
```

以上配置完成后，重启 Apache 服务器即可。

注意　如果用户的计算机上还有 IIS 服务器，那么可能会因为端口冲突而导致 Apache 无法正常开启。解决的办法是改变其中的一个端口号，或者停止 IIS 服务器。

1.2.2　在 Linux 下安装 Apache 服务器

在 Linux 下安装 Apache 需要到官方网站 http://www.apache.org 下载 Linux 下的 httpd2.2.8.tar.gz 压缩包。

首先需要打开 Linux 终端（Linux 系统中，几乎所有的软件都需要在终端下安装）。在 RedHat9 的界面中选择“主菜单”/“系统工具”命令，在弹出的菜单中选择“终端”命令。

在 Linux 下安装和配置 Apache 服务器的操作步骤如下：

（1）进入到 Apache 安装文件的目录下，如/usr/local/work。

```
cd /usr/local/work/
```

（2）解压安装包。完成后进入到 httpd2.2.8 目录中。

```
tar xfz httpd2.2.8.tar.gz
cd httd2.2.8
```

（3）建立 makefile，将 Apache 服务器安装到 user/local/Apache2 下。

```
./configure –prefix=/usr/local/Apache2 –enable-module=so
```

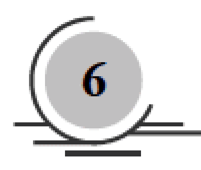
（4）编译文件。

```
make
```

（5）开始安装。

```
make install
```

（6）安装完成后，将 Apache 服务添加到系统启动项中，重启服务器。



```
/usr/local/Apache2/bin/Apachectl start >> /etc/rc.d/rc.local
/usr/local/Apache2/bin/Apachectl restart
```

（7）打开 Mozilla 浏览器，在地址栏中输入“http://localhost/”，按 Enter 键后如果看到如图 1.9 所示的页面，说明 Apache 服务器已安装成功。

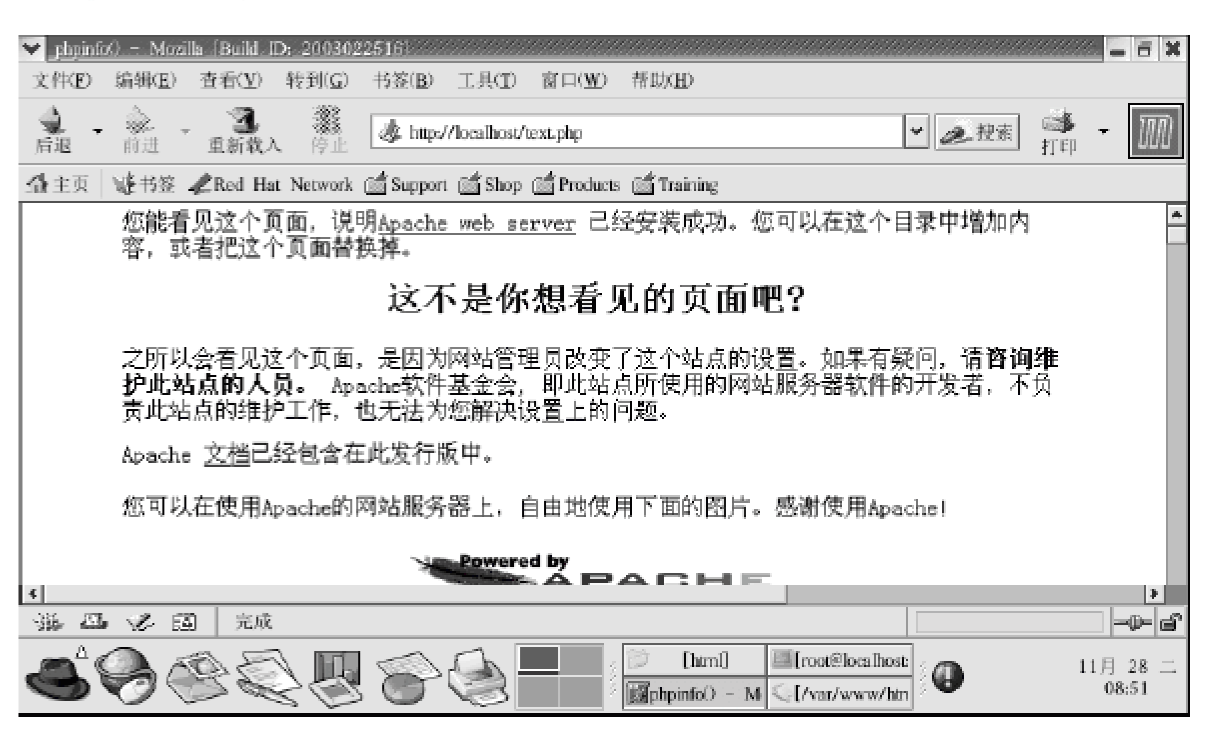


图 1.9　Linux 下的 Apache 服务器安装页面

1.3　PHP 的安装和配置

视频讲解：光盘\TM\Video\第 1 章\PHP 的安装和配置.exe

1.3.1　在 Windows 下安装 PHP

架设基于 PHP 的 Web 服务器，必须安装 PHP。由于 PHP 的代码公开，所以其升级速度较快。安装 PHP 之前应从其官方网站 http://www.php.net/下载最新版本的 PHP 安装程序 php-5.2.5-Win32.zip。

Apache 服务器顺利启动后，接下来安装 PHP5。在 Windows 下安装和配置 PHP 的操作步骤如下：

（1）将 PHP5 的安装文件 php-5.2.5-Win32.zip 解压到相应目录，如 C:\php、E:\php5 等。这里将其放到 E:\php5 目录下。目录结构如图 1.10 所示。

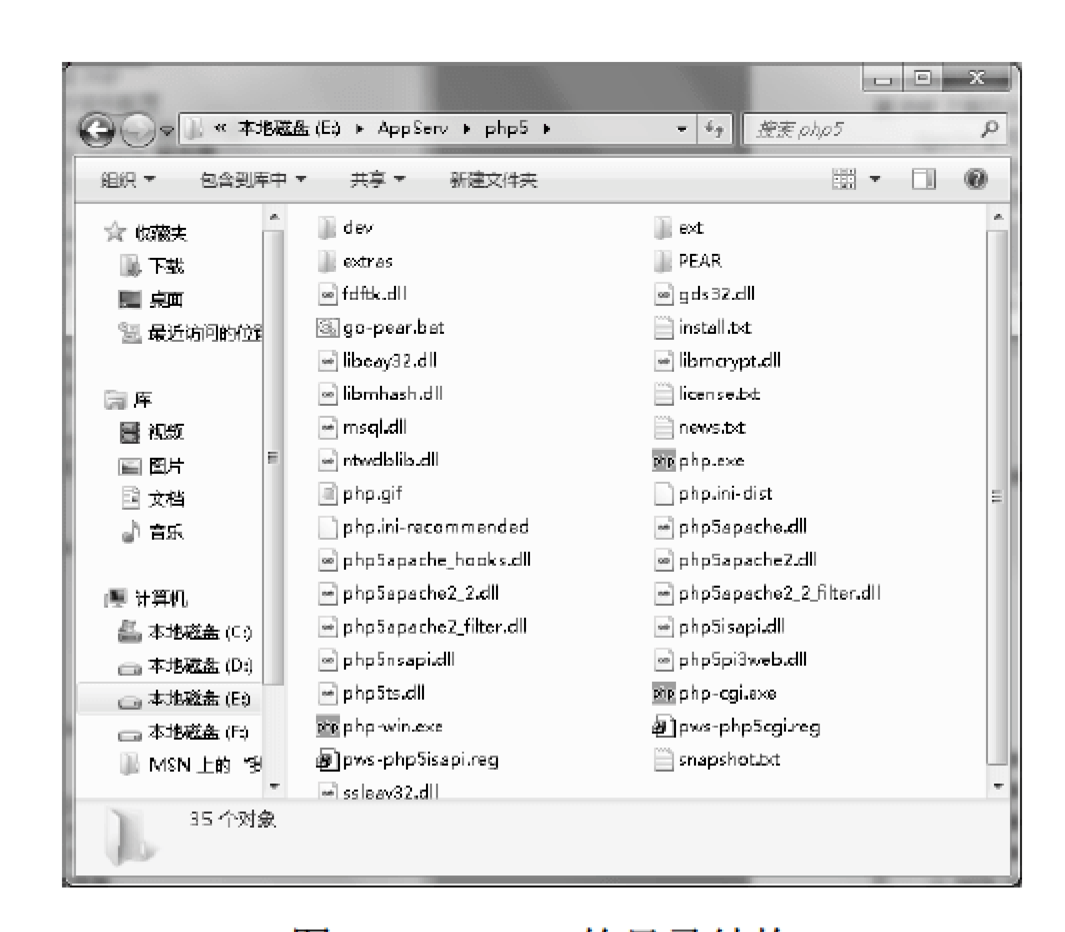


图 1.10　PHP5 的目录结构

（2）将该目录下的所有.dll 文件复制到系统盘 Windows\system32 目录下（Windows 2000 是在 winnt\system32 目录下）。

（3）将 php.ini-dist 文件复制到系统盘\Windows 目录下，并重新命名为 php.ini。

（4）打开 php.ini 文件并找到“extension_dir = "./"”，修改为“extension_dir = "e:/php5/ext"”。

（5）找到“;extension=php_mysql.dll”，将前面的分号“;”去掉。这样，PHP 即可支持 MySQL 数据库。

（6）PHP 配置完成以后，重新启动 Apache 服务器。

（7）编写一个 PHP 脚本文件，命名为 phpinfo.php，保存在 Apache 服务器的虚拟目录 D:\htdocs 下。PHP

脚本文件的代码如下：

```
<?php
phpinfo();                                    //获取 PHP 的配置信息
?>
```

（8）最后在浏览器的地址栏中输入“http://localhost/phpinfo.php”，如果显示 PHP 的版本相关信息，则说明 PHP 配置成功。

1.3.2　在 Linux 下安装 PHP

安装 PHP5 之前，需要首先查看 libxml 的版本号。如果 libxml 版本号小于 2.5.10，则需要先安装 libxml 高版本。安装 libxml 和 PHP5 的步骤如下（如果不需要安装 libxml，则直接跳到 PHP5 的安装步骤即可）：

（1）将 libxml 和 PHP5 复制到/usr/local/work/目录下，并进入到该目录。

```
cp php-5.2.5.tar.gz libxml2-2.6.26.tar.gz /usr/local/work
cd /usr/local/work
```

（2）分别将 libxml2 和 PHP 解压。

```
tar xfz libxml2-2.6.62.tar.gz
tar xfz PHP-5.2.5.tar.gz
```

（3）进入到 libxml2 目录，建立 makfile，将 libxml 安装到/usr/local/libxml2 下。

```
cd libxml2-2.6.62
./configure –prefix=/usr/local/libxml2
```

（4）编译文件。

```
makefile
```

（5）开始安装。

```
make install
```

（6）libxml2 安装完毕，开始安装 PHP5。进入到 php-5.2.5 目录下。

```
cd ../php-5.2.5
```

（7）建立 makefile。

```
./configure –with-apxs2=/usr/local/Apache2/bin/apxs
--with-mysql=/usr/local/mysql
--with-libxml-dir=/usr/local/libxml2
```

（8）开始编译。

```
make
```

（9）开始安装。

```
make install
```

（10）复制 php.ini-dist 或 php.ini-recommended 到/usr/local/lib 目录，并命名为 php.ini。

```
cp php.ini-dist /usr/local/lib/php.ini
```

（11）更改 httpd.conf 文件相关设置，该文件位于/usr/local/Apache2/conf 中。找到该文件中的如下指令行：

```
AddType application/x-gzip .gz .tgz
```

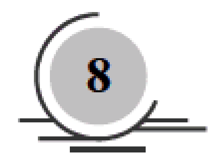
在该指令后加入如下指令：

```
AddType application/x-httpd-php .php
```

重新启动 Apache，并在 Apache 主目录下建立文件 phpinfo.php。

```
<?php
phpinfo();
?>
```

在 Mozilla 浏览器的地址栏中输入“http://localhost/phpinfo.php”，如果出现如图 1.11 所示的页面，说明 PHP 安装成功。



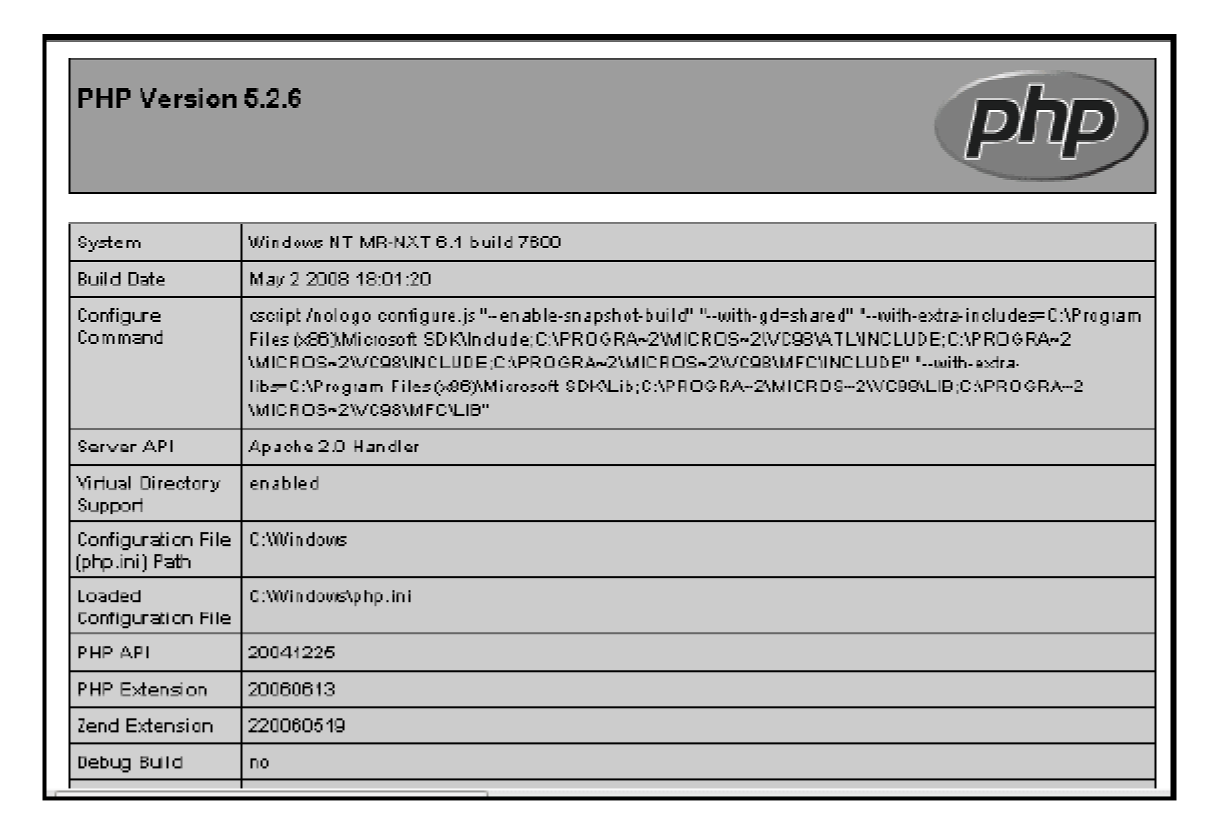


图 1.11　phpinfo 信息

1.4　应用组合包快速搭建 PHP 环境

视频讲解：光盘\TM\Video\第 1 章\应用组合包快速搭建 PHP 环境.exe

组合包，就是将 Apache、PHP 和 MySQL 等服务器软件和工具安装配置完成后打包处理。开发人员只要将已配置的套件解压到本地硬盘中即可使用，无须再另行配置。组合包实现了 PHP 开发环境的快速搭建。对于初学 PHP 的程序员，建议采用此方法搭建 PHP 的运行环境。组合包安装简单、速度较快、运行稳定，能使用户将精力更好地集中到 PHP 的学习中。

目前网上流行的组合包有十几种，安装方法基本相同。这里推荐 3 种组合包：EasyPHP、AppServ 和 XAMPP。

☑ EasyPHP 的下载地址为 http://www.easyphp.org/。

☑ AppServ 的下载地址为 http://www.appservnetwork.com/。

☑ XAMPP 的下载地址为 http://www.Apachefriends.org/。

建议新手使用 EasyPHP 或 AppServ，两者都是 Apache+MySQL+PHP 开发环境的。而 XAMPP 则相对复杂一些，不仅可以切换 PHP4 和 PHP5，还集成了 perl 开发环境、第三方扩展库等，并且对开发平台进行了优化和整理。如果对 PHP 的开发环境已经有了一定的了解，则推荐使用 XAMPP。

注意 要安装这些组合包，必须保证系统中未安装 Apache、PHP 和 MySQL，否则要先将这些软件卸载，再开始安装组合包。组合包的安装很简单，只要将程序解压或安装到指定目录就可以直接使用。

本节以 AppServ 组合包的 AppServ-win32-2.5.10 版本为例，重点讲解该组合包的安装和使用方法。

AppServ 是 PHP 网页架站工具组合包，是将网络上免费的架站资源重新包装成单一的安装程序。它提供了简易、快速的 PHP 运行环境的搭建机制，读者只需按照普通应用软件的安装方式就可以完成 Apache+MySQL+PHP+phpMyAdmin 的安装与配置工作。可以说，AppServ 是初学者的首选。

安装 AppServ 之前应从官方网站 http://www.appservnetwork.com 下载 AppServ-win32-2.5.10.exe 安装程序。在 Windows 下应用 AppServ 组合包快速搭建 PHP 开发环境的操作步骤如下：

（1）双击 AppServ-win32-2.5.10.exe 文件，打开如图 1.12 所示的 AppServ 启动页面。

（2）单击 Next 按钮，打开如图 1.13 所示的 AppServ 安装协议页面。

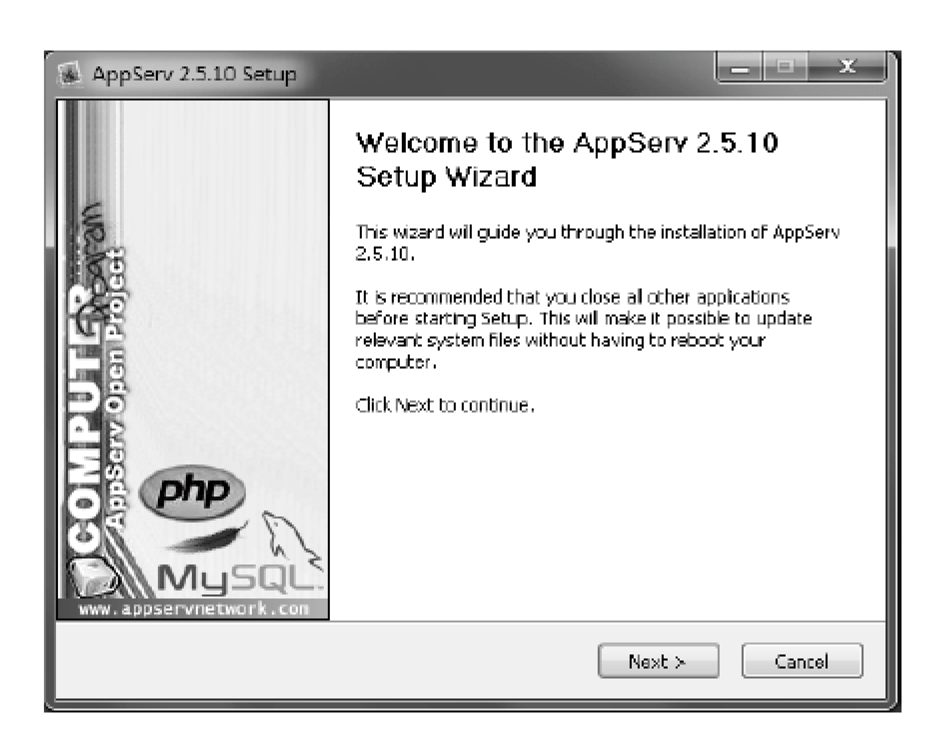


图 1.12　AppServ 启动页面

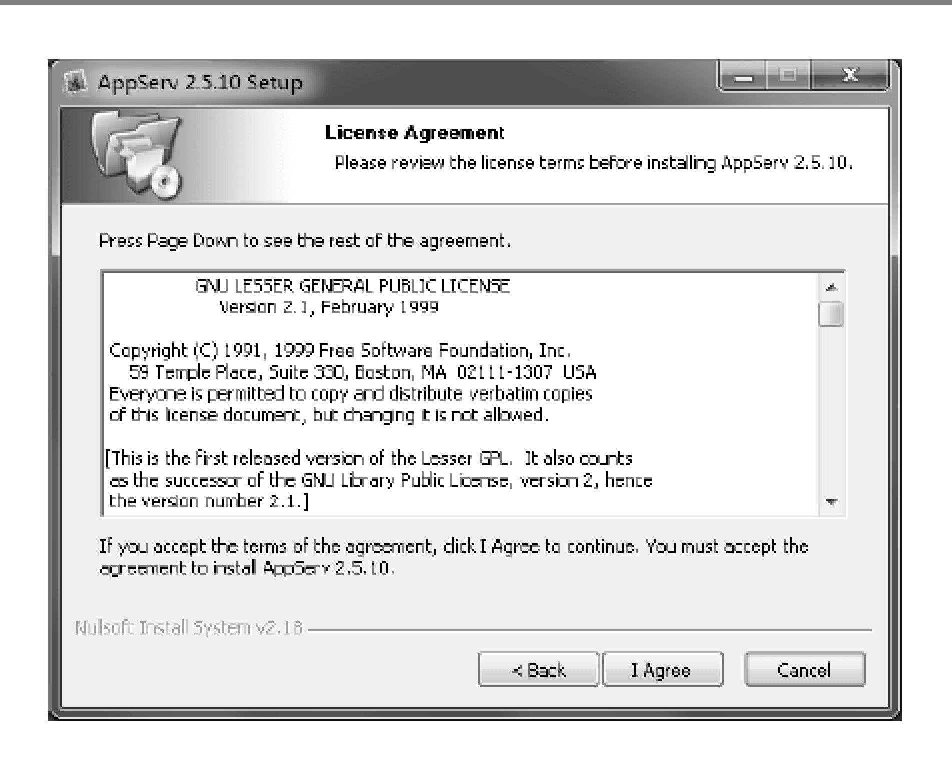


图 1.13　AppServ 安装协议页面

（3）单击 I Agree 按钮将打开如图 1.14 所示的页面。在该页面中可以设置 AppServ 的安装路径（默认安装路径一般为 C:\AppServ），AppServ 安装完成后，Apache、MySQL 和 PHP 都将以子目录的形式存储到该目录下。

（4）单击 Next 按钮将打开如图 1.15 所示的页面。在该页面中可以选择要安装的程序和组件（默认为全选状态）。



图 1.14　选择 AppServ 安装路径



图 1.15　AppServ 安装选项

（5）单击 Next 按钮，打开如图 1.16 所示的页面。该页面主要用于设置 Apache 的端口号。

（6）单击 Next 按钮，打开如图 1.17 所示的页面。该页面主要用于对 MySQL 数据库的 root 用户的登录密码及字符集进行设置。这里将字符集设置为 GB2312 Simplified Chinese，表示 MySQL 数据库的字符集将采用简体中文形式。

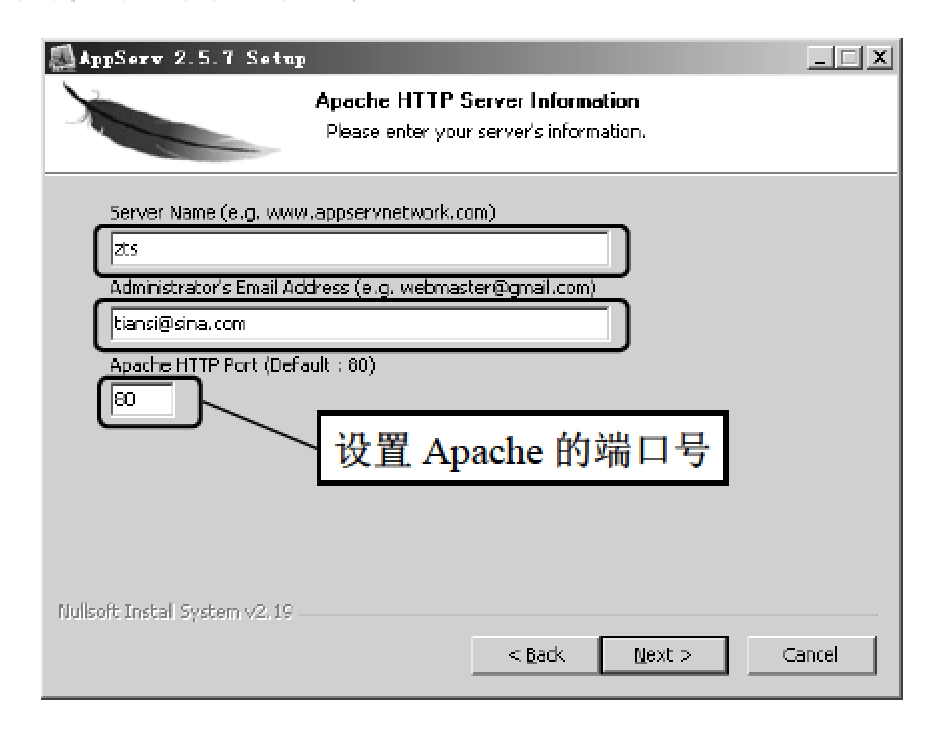


图 1.16　设置 Apache 端口号



图 1.17　设置 MySQL 登录密码及字符集

（7）单击 Install 按钮后开始安装，如图 1.18 所示。

（8）安装完成页面如图 1.19 所示。安装完成后，可以在开始菜单的 AppServ 相关操作列表中启动 Apache 及 MySQL 服务。

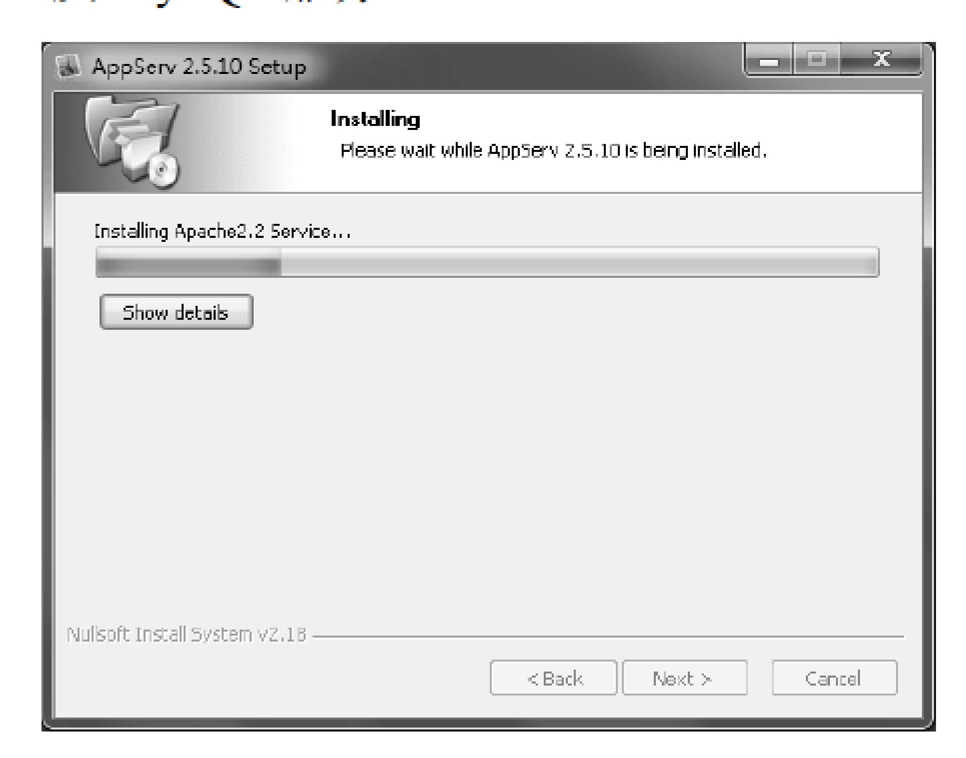


图 1.18　AppServ 安装进度页面

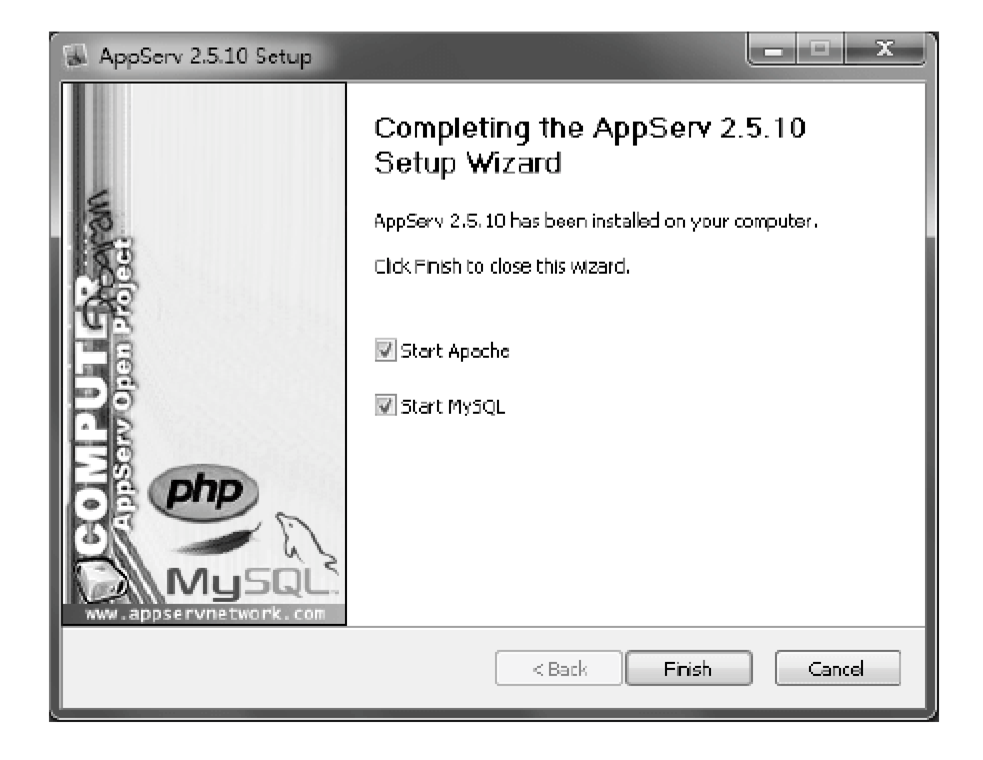


图 1.19　AppServ 安装完成页面

（9）安装好 AppServ 后，整个目录默认为 C:\AppServ，此目录下包含 4 个子目录，如图 1.20 所示，用户可以将所有网页文件存放到 www 目录下。

（10）打开浏览器，在 IE 浏览器的地址栏中输入“http://localhost”或者“http://127.0.0.1/”，如果打开如图 1.21 所示的页面，则说明 AppServ 安装成功。

AppServ
Apache2.2 —— Apache的存储目录
MySQL —— MySQL的存储目录
php5 —— PHP的存储路径
www —— 网页文件及phpMyAdmin的存储路径

图 1.20　AppServ 目录结构



图 1.21　AppServ 测试页

说明 如果在安装时设置 Apache 的端口号是 82，那么在 IE 浏览器的地址栏中输入“http://localhost:82/”或者“http://127.0.0.1:82/”来测试 AppServ 是否安装成功。

1.5　第一个 PHP 程序

视频讲解：光盘\TM\Video\第 1 章\第一个 PHP 程序.exe

1.5.1 使用 Adobe Dreamweaver 编写源程序

服务器环境配置完成后，接下来应用 Adobe Dreamweaver 开发工具来编写第一个 PHP 程序。

例 1.1 编写第一个 PHP 程序的目的是熟悉 PHP 的书写规则和 Adobe Dreamweaver 工具的基本使用方法。本实例很简单，就像初学大多数语言一样，输出一段欢迎信息。（**实例位置：光盘\TM\Instance\01\1.1**）

开发步骤如下：

（1）启动 Adobe Dreamweaver 编辑器，选择“文件”/“新建”命令，打开“新建文档”对话框，在“页面类型”列表框中选择 PHP 选项，如图 1.22 所示。

（2）单击“创建”按钮，即可成功创建一个动态的 PHP 页面。对新创建的页面，可以在“代码”视图中编辑 PHP 代码，也可以使用“设计”视图查看 HTML 效果。这里使用“代码”视图，并给该页面标题命名，如图 1.23 所示。标题显示在浏览器的左上角，稍后运行时就能看到效果。



图 1.22 “新建文档”对话框

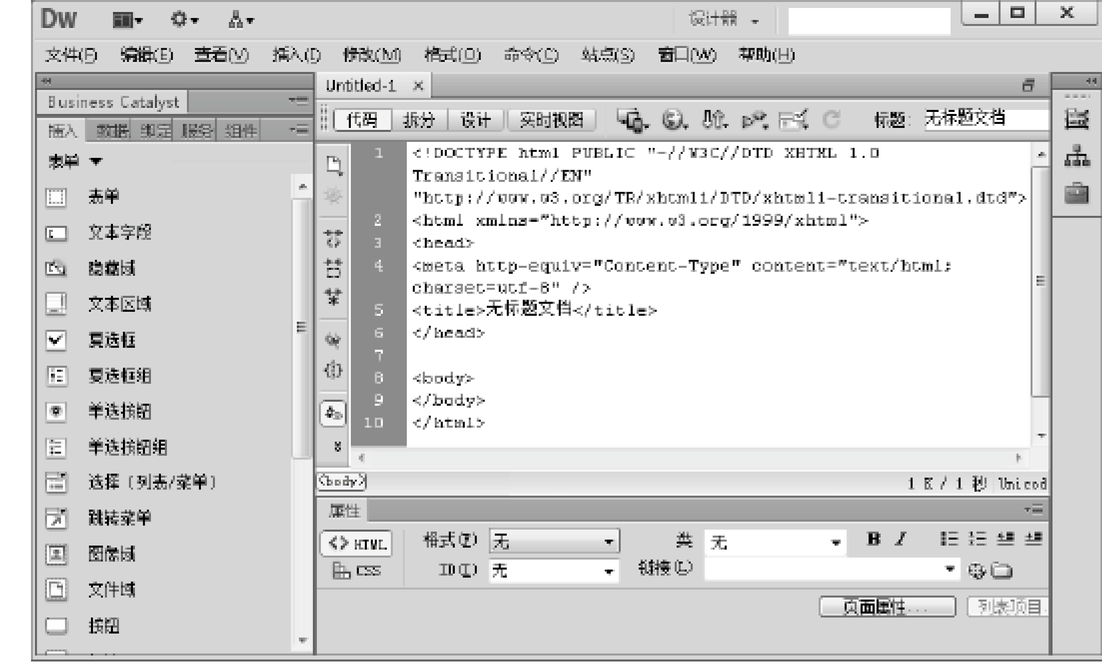


图 1.23 命名标题

（3）编写 PHP 代码。在<body>…</body>标签对中编写 PHP 代码段。代码如下：

```
<?php
    echo  "欢迎学习 PHP 语言！！";
?>
```

代码讲解如下：

☑ “<?php”和“?>”是 PHP 的标记符。在这对标记符中的所有代码都被当作 PHP 代码来处理。除了这种表示方法外，PHP 还可以使用 ASP 风格的“<%”和 SGML 风格的“<?…?>”等，在第 3 章中将会详细介绍。

☑ echo 语句是 PHP 中最常用的语句，主要用于将一个或多个字符串输出至网页，和 ASP 中的 response.write 以及 JSP 中的 out.print 意思相同，就是将紧跟在后面的字符串或者变量值显示在页面中。每行代码都以分号“;”结尾。

注意 在使用 echo 语句时，应该时刻牢记语句结束处的分号。在 PHP 中，分号是用来分隔语句的。丢失“;”是经常会犯的语法错误，同样也是在进行程序的错误处理时，首先应该注意的地方。

在 Adobe Dreamweaver 中输入的 PHP 脚本程序如图 1.24 所示。

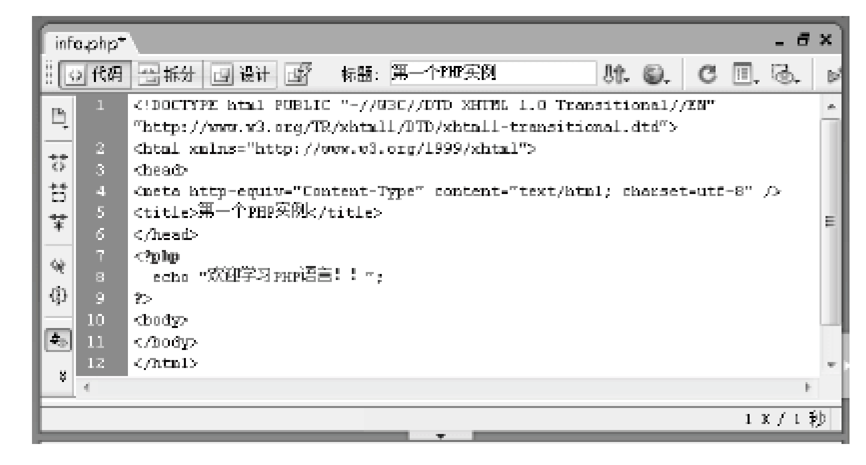


图 1.24 在开发工具中输入 PHP 脚本程序

（4）将 PHP 动态页保存到服务器指定的目录以便解析。本



章中服务器指定的目录为 E:\appserv\www\，则将本页保存到 E:\appserv\www\tm\01\1.1\路径下，命名为 info.php。

1.5.2　发布和运行 PHP 程序

打开 IE 浏览器，在地址栏中输入“http://127.0.0.1:82/TM/01/1.1/info.php”，按 Enter 键后即可查看 info.php 页的执行结果，如图 1.25 所示。

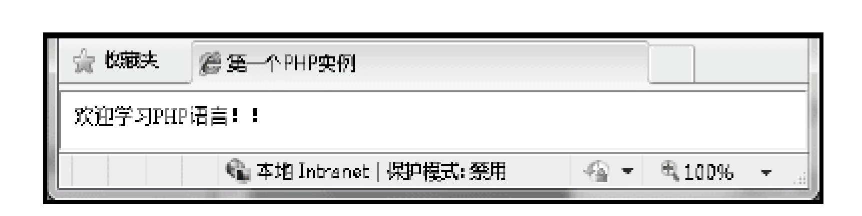


图 1.25　PHP 页面运行结果

1.6　环境安装常见问题

视频讲解：光盘\TM\Video\第 1 章\环境安装常见问题.exe

1.6.1　Apache 安装常见问题

1．解决 Apache 服务器端口冲突

IIS 服务器和迅雷的默认端口号为 80，同 Apache 服务器默认端口号相同。由于两者采用了相同的端口号 80，因此，在运行网页时就会发生冲突。

如果用户安装了 IIS 服务器，就需要修改 IIS 的默认端口，否则将导致 Apache 服务器无法正常工作。更改 IIS 的默认侦听端口 80，可以在 IIS 的管理器中进行设置，也可以停止 IIS 的服务。

如果用户安装并开启了迅雷软件，就需要关闭该软件，否则端口冲突将会导致运行 PHP 网页程序时出错。

用户也可以在安装 Apache 服务器时将默认的端口号进行更改，从而解决两个服务器或与其他软件共用一个端口号而产生冲突的问题。

2．更改 Apache 服务器默认存储的文件路径

Apache 服务器的核心配置文件是 httpd.conf，存放路径为“Apache 的安装路径\conf\”，用记事本程序打开该文件，定位到 DocumentRoot。语句如下：

```
DocumentRoot " D:/Webpage"
```

该语句用于指定网站路径，也就是主页放置的目录。可以使用默认路径，也可以任意指定。需要注意的是，语句的末尾不要加“/”。

同时还要定位到“<Directory "">”一行，在双引号中添加服务器的虚拟路径，这里要与“DocumentRoot”一行中设置相同。

```
<Directory " D:/Webpage ">
```



路径的分隔符在 Apache 服务器里写成“/”。

1.6.2　PHP 安装常见问题

1．PHP 的安装路径

安装文件的路径也要遵循一定的客观原则，为了避免在 Windows 和 Linux 间移植程序时带来的不便，

选择 D:\usr\local\php 的目录时要和在 Linux 下的安装目录相匹配。建议最好不要选择中间有空格的目录，如 E:\program Files\PHP，这样做会导致发生一些未知错误甚至崩溃。

2. 控制上传文件的大小

在网站开发的过程中，为了确保能够充分利用服务器的空间，禁止上传一些垃圾文件，给网站的维护带来不必要的麻烦，最好对上传文件的大小进行限制，将它控制在有效上传文件大小的范围内。如果要在 PHP 中实现小文件的上传（2MB 以下），那么无须对 php.ini 配置文件进行修改，使用默认参数即可。但如果想实现完美的上传功能，则一定要对 php.ini 进行一些修改。

Resource Limits，直译就是资源限制，包含 3 个参数。该区块不仅针对上传和下载，还可对全部的文件进行设置。各个参数含义及参数值说明如表 1.1 所示。

表 1.1　Resource Limits 块的参数说明

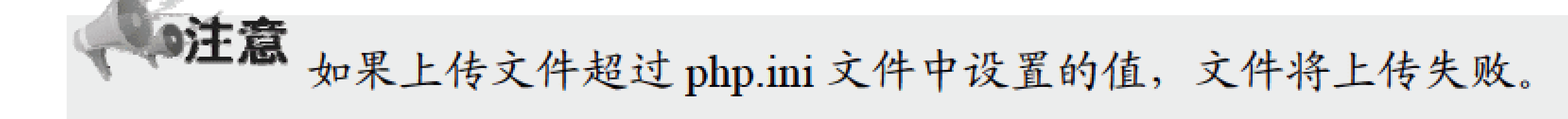
参　数	说　明
max_execution_time	每个脚本页面完成执行操作的最大时间，单位是秒。如果设为-1，说明没有限制
max_input_time	每个脚本页面处理请求数据的最大时间，单位是秒，也可以设为-1
memory_limit	一个脚本页所能够消耗的最大内存

File Uploads 块是专为文件上传设置的，包含 3 个参数，参数含义及参数值说明如表 1.2 所示。

表 1.2　File Uploads 块的参数说明

参　数	说　明
file_uploads	是否允许 HTTP 上传，默认为 On，即为开启，无须修改
upload_tmp_dir	文件上传时的临时存储目录。如果没指定就会用系统默认的临时文件夹
upload_max_filesize	允许上传的文件的最大值

如果想要上传更大的文件，就必须对上述 3 个参数值进行更改，更改后重新启动 Apache 服务器即可。



如果上传文件超过 php.ini 文件中设置的值，文件将上传失败。

1.6.3　MySQL 安装常见问题

在网站运作的过程中，各类错误均不可避免，当数据库连接失败时，除开启 MySQL 服务检测是否正常运行外，还可以检查 php.ini 文件是否配置正确，以支持 MySQL 服务。

打开 C:\Windows\目录下的 php.ini 文件，定位到如图 1.26 所示的代码位置。

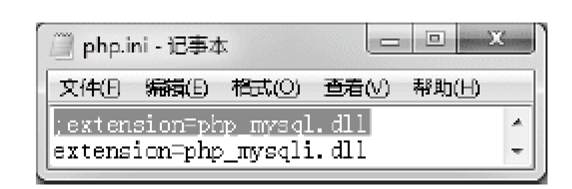
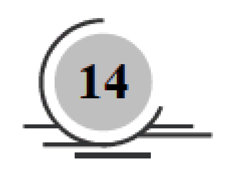


图 1.26　修改 php.ini 文件以支持 MySQL 数据库

将代码前面的分号删除，然后保存 php.ini 文件，最后重新启动 Apache 服务器，即可让 PHP 支持 MySQL 数据库。

在浏览器的地址栏中输入“http://127.0.0.1/phpinfo.php”或“http://localhost/phpinfo.php”，如果检索到 MySQL 服务，如图 1.27 所示，则说明 MySQL 服务正常运行。



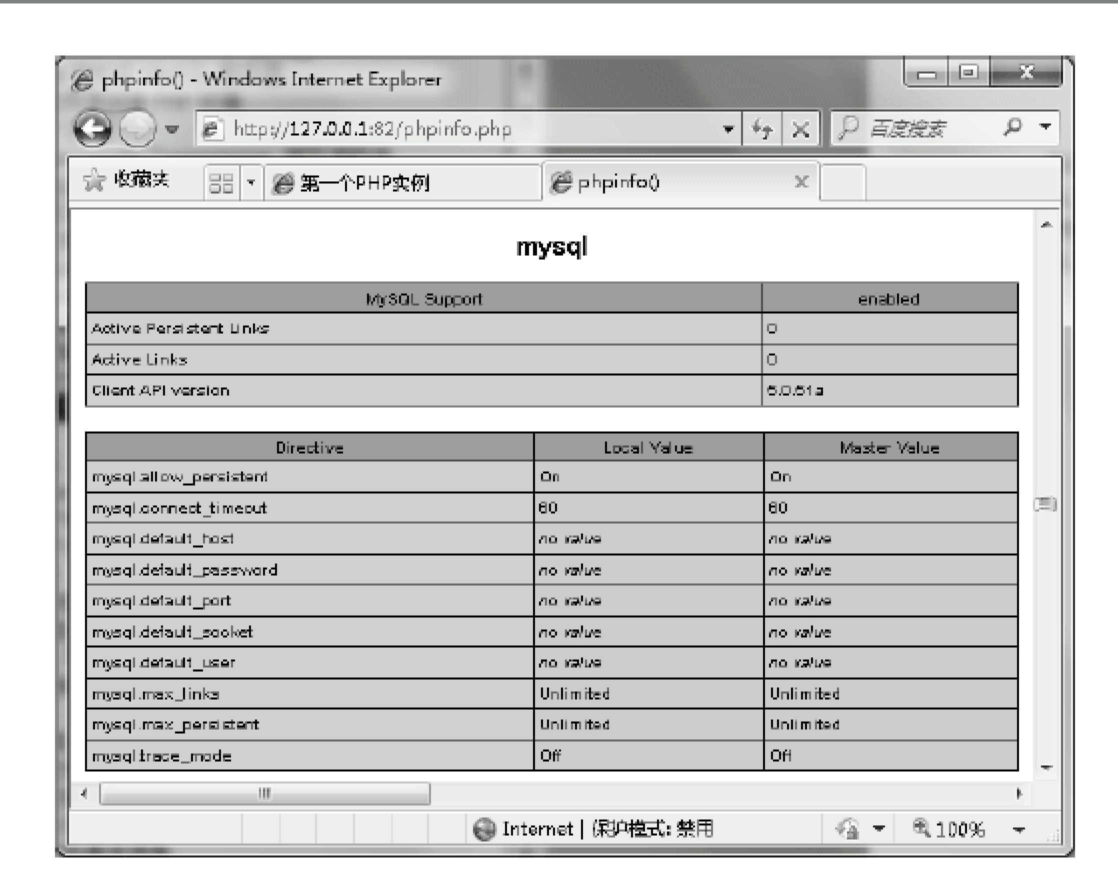


图 1.27 测试 MySQL 服务是否正常运行

1.7 实 战

例 1.2 应用 Adobe Dreamweaver 开发工具，通过调用 PHP 的 date()函数，动态显示系统时间。（**实例位置：光盘\TM\Instance\01\1.2**）

1．创建站点

编程的第一步就是创建 Web 站点，在 www 服务器的根目录下创建指定的 TM/01/1.2 文件夹，用来存储 PHP 文件和相关资源。

2．使用 Adobe Dreamweaver 创建 PHP 文件

启动 Adobe Dreamweaver 编辑器，选择“文件”/“新建”命令，打开“新建文档”对话框，选择“空白页”选项卡中的“页面类型”/“PHP 选项”，然后单击“创建”按钮，即可成功创建一个动态的 PHP 页面。

说明 将切割的图像嵌入到 PHP 页面中，并合理设计页面布局。

3．创建 PHP 标记

在<body>标记中使用“<?php ?>”标记符将 PHP 的脚本语言括起来，而这些 PHP 脚本在浏览器中并不可见，当用户在 IE 浏览器中输入 URL 地址“http://localhost/PHP Web/index.php”，服务器端接到请求后，会使用 PHP 的默认语言解释成标准的 HTML 格式的代码传送到用户的浏览器中。

技巧 解释生成的 HTML 代码，用户可以在 IE 浏览器中选择“查看”/“源文件”命令，打开程序的源代码，它是标准的 HTML 文件。

因为 PHP 代码在服务器端运行，在执行后才产生浏览器能识别的 HTML 语言，并将其传递给浏览器。

4．编写 PHP 代码

在<body>标记中输入如下代码：

```
<?php
    echo "<font color='#ffDFDF'>现在时刻北京时间：". date("Y-m-d H:i:s ")."</font>"
```

```
?>
```

代码解释如下：

在<body>…</body>标签对中添加“<?php”和“?>”标记符。第一行的“<?php”是 PHP 代码段的开始标记。第三行的“?>”是 PHP 代码段的结束标记。

第二行的 echo 语句用来输出字符串。其中，“…”代表将包含在该标记间的字符串设置指定的颜色（这里为粉色）。调用 date()函数来获取系统的当前时间，其中“Y-m-d H:i:s”是以指定的格式输出系统的当前日期和时间，并用字符串操作符“.”将多个字符串连接在一起进行输出。

注意 由于在 PHP 语言中默认设置的是标准的格林威治时间（即采用的是零时区），所以要获取本地当前的时间必须更改 PHP 语言中的时区设置。更改 PHP 语言中的时区设置有两种方法：（1）修改 php.ini 文件中的设置，找到[date]下的“;date.timezone =”选项，将该项修改为“date.timezone =Asia/Hong_Kong”，然后重新启动 Apache 服务器。

（2）在程序中，使用时间/日期函数前添加函数“date_default_timezone_set("Asia/Hong_Kong");”即可。

在 Adobe Dreamweaver 中输入 PHP 脚本程序，如图 1.28 所示。

5．存储文件

在创建好 PHP 文件后，选择“文件”/“保存”命令，在弹出的“另存为”对话框中选择已创建的 TM/01/1.2 文件夹作为 PHP 文件的保存位置，将文件命名为 index.php，然后单击“保存”按钮，如图 1.29 所示。



图 1.28　在开发工具中输入 PHP 脚本程序



图 1.29　“另存为”对话框

6．网站运行结果

打开 IE 浏览器，在地址栏中输入 URL 地址“http://127.0.0.1:82/tm/01/1.2/index.php”，按 Enter 键打开该页面，运行结果如图 1.30 所示。



图 1.30　打开网站页面

1.8　小　　结

本章主要介绍了 PHP 环境的安装与配置方法，读者应重点掌握。通过本章的学习，读者可以轻松解决环境配置的常见问题，并可以独立开发、发布和运行简单的 PHP 程序。对于本章讲解的在 Windows 和 Linux 系统上搭建 PHP 环境的方法，读者可以根据自身的需求有选择性地阅读。

1.9　学习成果检验

1．尝试开发一个页面，应用 echo 语句输出字符串“一定要努力学习 PHP 编程语言。”。**（答案位置：光盘\TM\Instance\01\1.3）**

2．尝试开发一个页面，应用 echo 语句输出一条最新公告信息，要求公告后面添加发布日期和时间，输出效果如图 1.31 所示。**（答案位置：光盘\TM\Instance\01\1.4）**



图 1.31　发布最新公告信息

3．尝试开发一个页面，使用 echo 语句输出一个 4×3 像素大小的表格。**（答案位置：光盘\TM\Instance\01\1.5）**

第 2 章

PHP 语言基础

（视频讲解：172 分钟）

通过第 1 章的学习，相信读者对 PHP 的概念和如何搭建 PHP 开发环境已有了一个全面的了解，本章将学习 PHP 的语言基础。无论是初出茅庐的“菜鸟”，还是资历深厚的“高手”，没有扎实的基础做后盾是不行的。PHP 的特点是易学、易用，但这并不代表随随便便就可以熟练掌握。随着知识的深入，PHP 会越来越难学，基础的重要性也就越加明显。掌握了基础，就等于有了坚固的地基，才有可能“万丈高楼平地起”。

通过阅读本章内容，你可以：

- 了解 PHP 的标记风格
- 了解 PHP 的数据类型
- 了解 PHP 的常量应用
- 了解 PHP 的变量应用
- 了解 PHP 运算符
- 了解 PHP 的自定义函数
- 掌握 PHP 的输出语句

2.1　PHP 语法基础

2.1.1　PHP 标记风格

视频讲解：光盘\TM\Video\第 2 章\PHP 标记风格.exe

PHP 和其他几种 Web 语言一样，其标记符能够让 Web 服务器识别 PHP 代码的开始和结束，两个标记之间的所有文本都会被解释为 PHP，而标记之外的任何文本都会被认为是普通的 HTML，这就是 PHP 标记的作用。PHP 标记符风格迥异，按照风格的不同可划分为以下 4 种。

（1）标准风格

```
<?php
    echo "标准风格的 PHP 标记";
?>
```

这是本书中使用的标记风格，也是推荐读者使用的标记风格。

（2）脚本风格

```
<script  language="php">
    echo '这是脚本风格的标记';
</script>
```

在 XHTML 或者 XML 中推荐使用这种标记风格，它符合 XML 语言规范的写法。

（3）简短风格

```
<?
    echo "简短风格的标记" ;
?>
```

这种标记风格最为简单，输入字符最少，但想要使用它，必须要更改配置文件 php.ini。这种标记风格不推荐使用。

（4）ASP 风格

```
<%
echo "ASP 风格的标记";
%>
```

注意 如果使用简短风格“<? ?>”和 ASP 风格“<% %>”，需要分别在配置文件 php.ini 中做如下设置。打开系统盘 Windows（以 Windows 2003 Server 操作系统为例）文件夹下的 php.ini 文件，将如下代码段中的“Off”改为“On”。更改后的代码如下：

```
short_open_tag = On
asp_tags = On
```

保存修改后的 php.ini 文件，然后重新启动 Apache 服务器，即可支持这两种标记风格。

2.1.2　PHP 注释应用

视频讲解：光盘\TM\Video\第 2 章\PHP 注释应用.exe

注释即代码的解释和说明，一般添加到代码的上方或代码的尾部（添加到代码的尾部时，代码和注释之间以 Tab 键进行分隔，以方便阅读程序），用来说明代码或函数的编写人、用途、时间等。注释不会影

响到程序的执行，因为注释部分在执行时会被解释器忽略。

PHP 支持 3 种风格的程序注释。

☑ //单行注释

```
<?php
    echo 'PHP 开发实战宝典';                    //输出字符串（但单行标记后的注释内容不被输出）
?>
```

☑ /*…*/ 多行注释

```
<?php
    /*多行
    注释内容
    不被输出
    */
    echo '只会看到这句话。';
?>
```

注意 多行注释不允许进行嵌套操作。

☑ Shell 风格的注释

```
<?php
    echo '这是 Shell 脚本风格的注释';              #这里的内容是看不到的
?>
```

注意 在单行注释里的内容不要出现“?>”的标志，因为解释器会认为 PHP 脚本结束，而去执行“?>”后面的代码。例如：

```
<?php
    echo '这样会出错的！！！'                    #不会看到?>会看到
?>
```

结果为：这样会出错的!!! 会看到?>

2.2　PHP 的数据类型

视频讲解：光盘\TM\Video\第 2 章\PHP 的数据类型.exe

PHP 一共支持 8 种原始类型，包括 4 种标量类型，即 boolean（布尔型）、integer（整型）、float/double（浮点型）和 string（字符串型）；两种复合类型，即 array（数组）和 object（对象）；两种特殊类型，即 resource（资源）与 null。

2.2.1　标量数据类型

标量数据类型是数据结构中最基本的单元，只能存储一个数据。PHP 中标量数据类型包括 4 种，如表 2.1 所示。

表 2.1　标量数据类型

类　　型	说　　明
boolean（布尔型）	这是最简单的类型。只有两个值，真（true）和假（false）
string（字符串型）	字符串就是连续的字符序列，可以是计算机所能表示的一切字符的集合
integer（整型）	整型数据类型只能包含整数。这些数据类型可以是正数或负数
float（浮点型）	浮点数据类型用于存储数字，和整型不同的是它有小数位

1．布尔型（boolean）

布尔型是 PHP 中较为常用的数据类型之一，它保存一个 true 值或者 false 值，其中 true 和 false 是 PHP 的内部关键字。设定一个布尔型的变量，只需将 true 或者 false 赋值给变量即可。

例 2.1　通常布尔型变量都是应用在条件或循环语句的表达式中。下面在 if 条件语句中判断变量$b 中的值是否为 true，如果为 true，则输出“变量$b 为真!”，否则输出“变量$b 为假!!”。

实例代码如下：**（实例位置：光盘\TM\Instance\02\2.1）**

```
<?php
     $b = true;                              //声明一个 boolean 类型变量，赋初值为 true
     if($b == true)                          //判断变量$b 是否为真
          echo '变量$b 为真!';               //如果为真，输出“变量$b 为真!”的字样
     else
          echo '变量$b 为假!!';              //如果为假，则输出“变量$b 为假!!”的字样
?>
```

结果为：变量$b 为真!

注意　在 PHP 中不是只有 false 值才为假的，在一些特殊情况下 boolean 值也被认为是 false。这些特殊情况为：0、0.0、"0"、空字符串（""）、只声明没有赋值的数组等。

说明　美元符号$是变量的标识符，所有变量都是以$开头的，无论是声明变量还是调用变量，都应使用$。

2．字符串型（string）

字符串是连续的字符序列，由数字、字母和符号组成。字符串中的每个字符只占用一个字节。在 PHP 中，有 3 种定义字符串的方式，分别是单引号（'）、双引号（"）和界定符（<<<）。

单引号和双引号是经常被使用的定义方式。定义格式如下：

```
<?php
$a ='字符串';
?>
```

或

```
<?php
$a ="字符串";
?>
```

两者的不同之处在于，双引号中所包含的变量会自动被替换成实际数值，而单引号中包含的变量则按普通字符串输出。

例 2.2　下面的实例分别应用单引号和双引号来输出同一个变量，其输出结果完全不同，双引号输出的是变量的值，而单引号输出的是字符串“$i”。**（实例位置：光盘\TM\Instance\02\2.2）**

实例代码如下：

```
<?php
    $i = '只会看到一遍';                          //声明一个字符串变量
    echo "$i";                                    //用双引号输出
    echo "<p>";                                   //输出段标记
    echo '$i';                                    //用单引号输出
?>
```

运行结果如图 2.1 所示。

两者之间另一处不同点是对转义字符的使用。使用单引号时，只要对单引号（'）进行转义即可，但使用双引号（"）时，还要注意“"”、“$”等字符的使用。这些特殊字符都要通过转义符（\）来显示。常用的转义字符如表 2.2 所示。

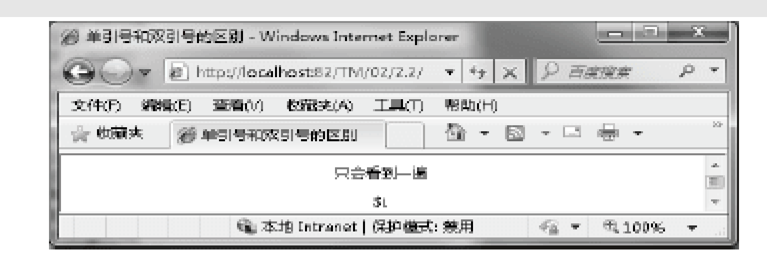


图 2.1　单引号和双引号的区别

表 2.2　转义字符

转 义 字 符	输　　出
\n	换行（LF 或 ASCII 字符 0x0A（10））
\r	回车（CR 或 ASCII 字符 0x0D（13））
\t	水平制表符（HT 或 ASCII 字符 0x09（9））
\\	反斜杠
\$	美元符号
\'	单引号
\"	双引号
\[0-7]{1,3}	此正则表达式序列匹配一个用八进制符号表示的字符，如\467
\x[0-9A-Fa-f]{1,2}	此正则表达式序列匹配一个用十六进制符号表示的字符，如\x9f

\n 和\r 在 Windows 系统中没有什么区别，都可以当作回车符。但在 Linux 系统中则是两种效果，\n 表示换到下一行，却不会回到行首；而\r 表示光标回到行首，但仍然在本行。如果读者使用 Linux 操作系统，可以尝试一下。

注意　如果对非转义字符使用了“\”，那么在输出时，“\”也会跟着一起被输出。

说明　在定义简单的字符串时，使用单引号是一个更加合适的处理方式。如果使用双引号，PHP 将花费一些时间来处理字符串的转义和变量的解析。因此，在定义字符串时，如果没有特别的要求，应尽量使用单引号。

界定符（<<<）是从 PHP 4.0 开始支持的。在使用时后接一个标识符，然后是字符串，最后是同样的标识符结束字符串。界定符的格式如下：

```
$string = <<< str
要输出的字符串。
str
```

其中 str 为指定的标识符。

例 2.3　下面使用界定符输出变量中的值，可以看到，它和双引号没什么区别，包含的变量也被替换成实际数值。**（实例位置：光盘\TM\Instance\02\2.3）**

实例代码如下：

```
<?php
    $i = '显示该行内容';                                  //声明变量$i
```



```
    echo <<<std                                          //界定符开始
    这和双引号没有什么区别，\$i 同样可以被输出出来。<p>        //输出字符串
    \$i 的内容为：$i                                       //输出变量$i
std;                                                     //界定符结束
?>
```

运行结果如图 2.2 所示。

注意 结束标识符必须单独另起一行，并且不允许有空格。在标识符前后有其他符号或字符，也会发生错误。例 2.3 中的注释部分在练习时一定不要输入，否则将出现“Parse error: parse error, unexpected T_SL in E:\AppServ\www\tm\02\2.3\index.php on line…”的错误提示。

3．整型（integer）

整型数据类型只能包含整数。在 32 位的操作系统中，有效的范围是-2147483648～+2147483647。整型数可以用十进制、八进制和十六进制来表示。如果用八进制，数字前面必须加 0，如果用十六进制，则需要加 0x。

注意 如果在八进制中出现了非法数字（8 和 9），则后面的数字会被忽略掉。

例 2.4　本例分别输出八进制、十进制和十六进制的结果。**（实例位置：光盘\TM\Instance\02\ 2.4）**

实例代码如下：

```
<?php
    $str1 = 1234567890;                              //声明一个十进制的整数
    $str2 = 0x1234567890;                            //声明一个十六进制的整数
    $str3 = 01234567890;                             //声明一个八进制的整数
    $str4 = 01234567;                                //声明另一个八进制的整数
    echo '数字 1234567890 不同进制的输出结果：<p>';
    echo '10 进制的结果是：'.$str1.'<br>';              //输出十进制整数
    echo '16 进制的结果是：'.$str2.'<br>';              //输出十六进制整数
    echo '8 进制的结果是：';
if($str3 == $str4){                                  //判断$str3 和$str4 的关系
        echo '$str3 = $str4 = '.$str3;               //如果相等，输出变量值
    }else{
        echo '$str3 != str4';                        //如果不相等，输出“$str3 != $str4”
    }
?>
```

运行结果如图 2.3 所示。

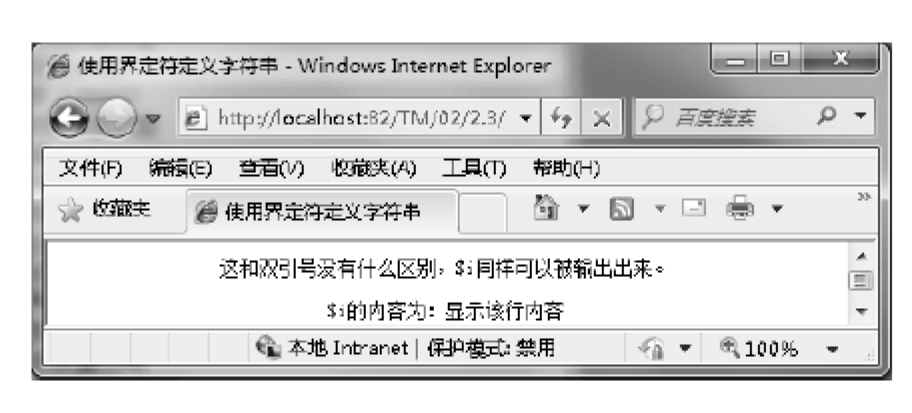


图 2.2　使用界定符定义字符串

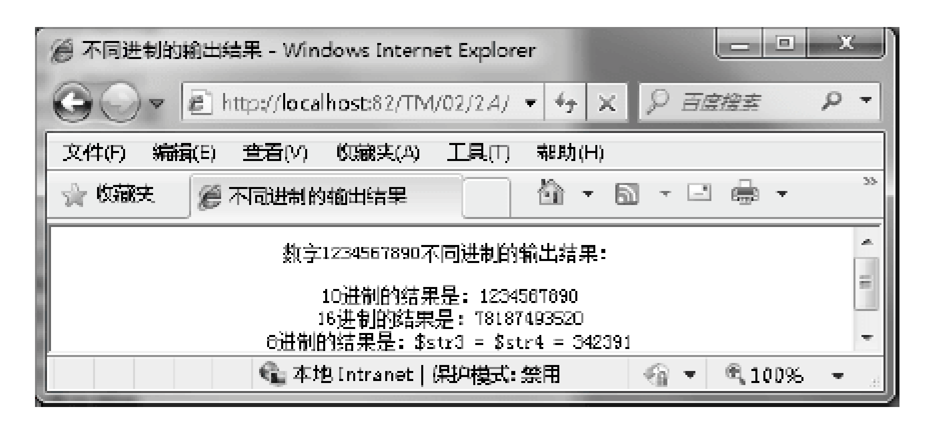


图 2.3　不同进制的输出结果

注意 如果给定的数值超出了 int 型所能表示的最大范围，将会被当作 float 型处理，这种情况称为整数溢出。同样，如果表达式的最后运算结果超出了 int 型的范围，也会返回 float 型。

4．浮点型（float）

浮点数据类型可以用来存储数字，也可以保存小数。它提供的精度比整数大得多。在 32 位的操作系统中，有效的范围是 1.7E-308～1.7E+308。在 PHP 4 以前的版本中，浮点型的标识为 double，也叫做双精度浮点数，两者没有区别。

浮点型数据默认有两种书写格式，一种是标准格式：

```
3.1415
-35.8
```

还有一种是科学计数法格式：

```
3.58E1
849.72E-3
```

例 2.5 本例中输出圆周率的近似值。用 3 种书写方法：圆周率函数、传统书写格式和科学记数法，最后显示在页面上的效果都一样。**（实例位置：光盘\TM\Instance\02\2.5）**

实例代码如下：

```
<?php
echo '圆周率的 3 种书写方法：<p>';
echo '第一种：pi() = '. pi() .'<p>';                                    //调用 pi()函数输出圆周率
echo '第二种：3.14159265359 = '. 3.14159265359 .'<p>';                  //传统书写格式的浮点数
echo '第三种： 314159265359E-11 = '. 314159265359E-11 .'<p>';           //科学计数法格式的浮点数
?>
```

运行结果如图 2.4 所示。

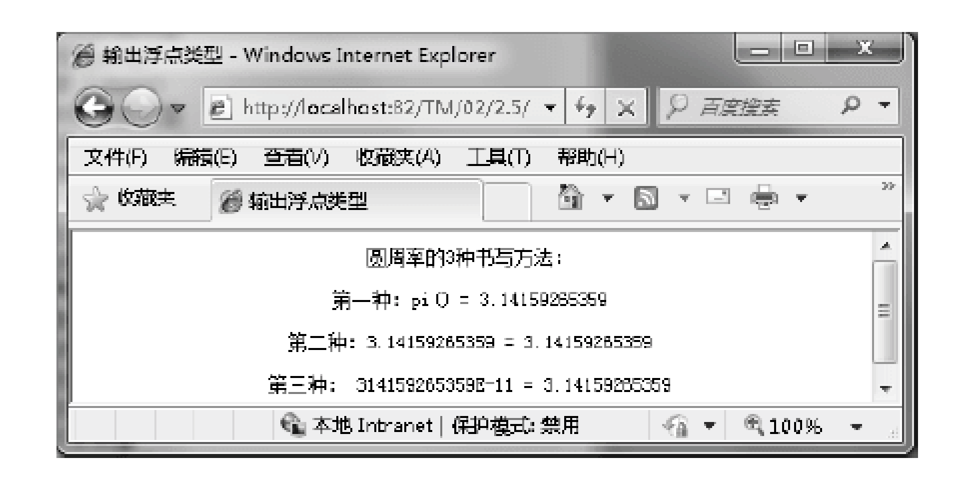


图 2.4 输出浮点类型

注意 浮点型的数值只是一个近似值，所以要尽量避免浮点型数值之间比较大小，因为最后的结果往往是不准确的。

2.2.2 复合数据类型

复合数据类型包括两种，即数组和对象，如表 2.3 所示。

表 2.3 复合数据类型

类　型	说　明
array（数组）	一组类型相同的变量的集合
object（对象）	对象是类的实例，使用 new 命令来创建

1．数组（array）

数组是一组数据的集合，它把一系列数据组织起来，形成一个可操作的整体。数组中可以包括很多数据，如标量数据、数组、对象、资源以及 PHP 中支持的其他语法结构等。

数组中的每个数据称为一个元素，元素包括索引（键名）和值两个部分。元素的索引可以由数字或字符串组成，元素的值可以是多种数据类型。定义数组的语法格式如下：

```
$array = ('value1',' value2 '……)
```

或

```
$array[key] = 'value'
```

或

```
$array = array(key1 => value1, key2 => value2……)
```

其中，参数 key 是数组元素的下标，value 是数组下标所对应的元素。以下几种都是正确的格式：

```
$arr1 = array('This','is','a','example');
```



```
$arr2 = array(0 => 'php', 1=>'is', 'the' => 'the', 'str' => 'best ');
$arr3[0] = 'tmpname';
```

声明数组后，数组中的元素个数还可以自由更改。只要给数组赋值，数组就会自动增加长度。在第 7 章中会详细介绍数组的使用、取值以及数组的相关函数。

2. 对象（object）

编程语言所应用到的方法有两种：面向过程和面向对象。在 PHP 中，用户可以自由使用这两种方法。在第 8 章中将对面向对象的技术进行详细讲解。

2.2.3　特殊数据类型

特殊数据类型包括资源和空值两种，如表 2.4 所示。

表 2.4　特殊数据类型

类　　型	说　　明
resource（资源）	资源是一种特殊变量，又叫做句柄，保存到外部资源的一个引用。资源是通过专门的函数来建立和使用的
null（空值）	特殊的值，表示变量没有值，唯一的值就是 null

1. 资源（resource）

资源类型是 PHP 4 引进的。关于资源的类型，可以参考 PHP 手册后面的附录，里面有详细的介绍和说明。

在使用资源时，系统会自动启用垃圾回收机制，释放不再使用的资源，避免内存消耗殆尽。因此，资源很少需要手工释放。

2. 空值（null）

空值，顾名思义，表示没有为该变量设置任何值，另外，空值（null）不区分大小写，null 和 NULL 效果是一样的。被赋予空值的情况有以下 3 种：还没有赋任何值、被赋值 null、被 unset()函数处理过的变量。

例 2.6　下面来看一个具体实例。字符串 string1 被赋值为 null，string2 根本没有声明和赋值，所以也输出 null，最后的 string3 虽然被赋予了初值，但被 unset()函数处理后，也变为 null 型。unset()函数的作用就是从内存中删除变量。**（实例位置：光盘\TM\Instance\02\2.6）**

实例代码如下：

```
<?php
echo "变量(\$string1)直接赋值为 null：";
$string1 = null;                                    //变量$string1 被赋空值
$string3 = "str";                                   //变量$string3 被赋值 str
if(is_null($string1))                               //判断$string1 是否为空
	echo "string1 = null";
echo "<p>变量(\$string2)未被赋值：";
if(is_null($string2))                               //判断$string2  是否为空
	echo "string2 = null";
echo "<p>被 unset()函数处理过的变量(\$string3)：";
unset($string3);                                    //释放$string3
if(is_null($string3))                               //判断$string3 是否为空
	echo "string3 = null";
?>
```

运行结果如图 2.5 所示。

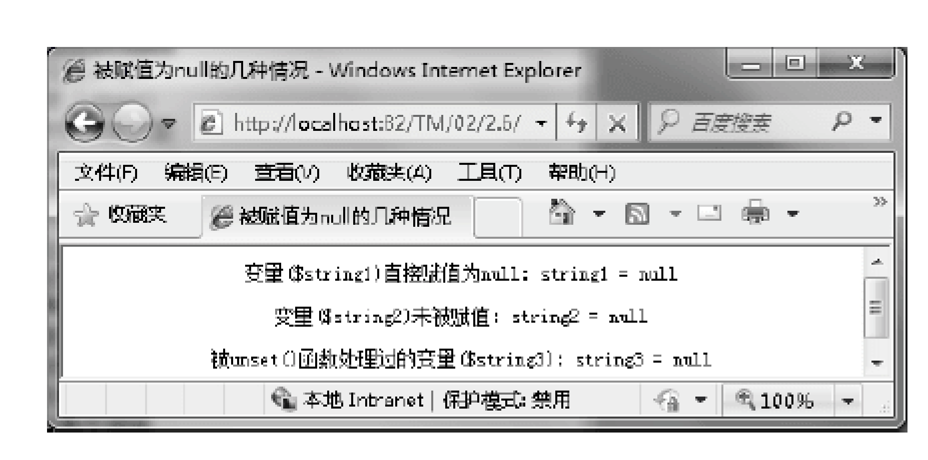


图 2.5　被赋值为 null 的几种情况



说明 is_null()函数是判断变量是否为 null，该函数返回一个 boolean 型，如果变量为 null，则返回 true，否则返回 false。unset()函数用来销毁指定的变量。

注意 从 PHP 4 开始，unset()函数就不再有返回值，所以不要试图获取或输出 unset()。

2.2.4　转换数据类型

虽然 PHP 是弱类型语言，但有时仍然需要用到类型转换。PHP 中的类型转换和 C 语言一样，非常简单，只需在变量前加上用括号括起来的类型名称即可。允许转换的类型如表 2.5 所示。

表 2.5　类型强制转换

转换操作符	转 换 类 型	举　　例
(boolean)	转换成布尔型	(boolean)$num、(boolean)$str
(string)	转换成字符型	(string)$boo、(string)$flo
(integer)	转换成整型	(integer)$boo、(integer)$str
(float)	转换成浮点型	(float)$str、(float)$str
(array)	转换成数组	(array)$str
(object)	转换成对象	(object)$str

注意 在进行类型转换的过程中应该注意以下内容：转换成 boolean 型时，null、0 和未赋值的变量或数组会被转换为 false，其他的为真；转换成整型时，布尔型的 false 转换为 0，true 转换为 1，浮点型的小数部分被舍去，字符型如果以数字开头就截取到非数字位，否则输出 0。

类型转换还可以通过 settype()函数来完成，该函数可以将指定的变量转换成指定的数据类型。

```
bool settype ( mixed var, string type )
```

参数 var 为指定的变量；参数 type 为指定的类型，有 7 个可选值，即 boolean、float、integer、array、null、object 和 string。如果转换成功则返回 true，否则返回 false。

当字符串转换为整型或浮点型时，如果字符串是以数字开头的，就会先把数字部分转换为整型，再舍去后面的字符串；如果数字中含有小数点，则会取到小数点前一位。

例 2.7　本实例将使用上面的两种方法将指定的字符串进行类型转换，比较两种方法之间的不同。（**实例位置：光盘\TM\Instance\02\2.7**）

实例代码如下：

```
<?php
$num = '2.1415926r*r';                              //声明一个字符串变量
echo '使用(integer)操作符转换变量$num 类型：';
echo (integer)$num;                                 //使用 integer 转换类型
echo '<p>';
echo '输出变量$num 的值：'.$num;                    //输出原始变量$num
```

```
echo '<p>';
echo '使用 settype 函数转换变量$num 类型：';
echo settype($num,'integer');                    //使用 settype 函数转换类型
echo '<p>';
echo '输出变量$num 的值：'.$num;                  //输出原始变量$num
?>
```

运行结果如图 2.6 所示。

可以看到，使用 integer 操作符能直接输出转换后的变量类型。并且原变量不发生任何变化。而使用 settype()函数返回的是 1，也就是 true，而原变量被改变了。在实际应用中，可根据情况自行选择转换方式。

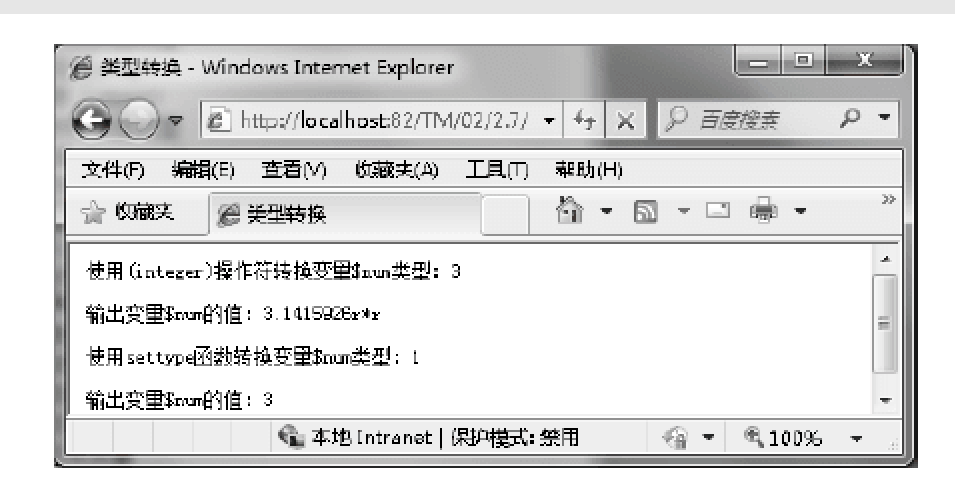


图 2.6　类型转换

2.2.5　检测数据类型

PHP 还内置了检测数据类型的系列函数，可以对不同类型的数据进行检测，判断其是否属于某个类型，如果符合则返回 true，否则返回 false。检测数据类型的函数如表 2.6 所示。

表 2.6　检测数据类型

函　　数	检 测 类 型	举　　例
is_bool	检查变量是否是布尔类型	is_bool(true)、is_book(false)
is_string	检查变量是否是字符串类型	is_string('string')、is_string(1234)
is_float/is_double	检查变量是否为浮点类型	is_float(3.1415)、is_float('3.1415))
is_integer/is_int	检查变量是否为整数	is_integer(34)、is_integer('34')
is_null	检查变量是否为 null	is_null(null)
is_array	检查变量是否为数组类型	is_array($arr)
is_object	检查变量是否是一个对象类型	is_object($obj)
is_numeric	检查变量是否为数字或由数字组成的字符串	is_numeric('5')、is_numeric('bccd110')

例 2.8　由于检测数据类型的函数的功能和用法都是相同的，下面使用 is_numeric()函数来检测变量中的数据是否是数字，从而了解并掌握 is 系列函数的用法。（**实例位置：光盘\TM\Instance\02\2.8**）

实例代码如下：

```
<?php
    $boo = "043112345678";                          //声明一个全由数字组成的字符串变量
    if(is_numeric($boo))                            //判断该变量是否由数字组成
        echo "Yes,the \$boo a phone number：$boo!"; //如果是，输出该变量
    else
        echo "Sorry,This is an error!";             //否则输出错误语句
?>
```

结果为：Yes,the $boo a phone number：043112345678。

2.3　PHP 的常量应用

视频讲解：光盘\TM\Video\第 2 章\ PHP 的常量应用.exe

常量可以理解为用于存储不经常改变的数据信息的量。常量的值被确定后，在程序的整个执行期间内，

这个值都有效，并且不可再次对该常量进行赋值。本节介绍 PHP 的常量，包括常量的声明和使用，以及预定义常量。

2.3.1 声明和使用常量

一个常量由英文字母、下划线和数字组成，但数字不能作为首字母出现。

在 PHP 中使用 define()函数来定义常量，该函数的语法格式为：

```
define(string constant_name,mixed value,case_sensitive=true)
```

☑ constant_name：必选参数，常量名称，即标识符。

☑ value：必选参数，常量的值。

☑ case_sensitive：可选参数，指定是否大小写敏感，设定为 true 表示不敏感。默认状态表示大小写敏感。

获取常量值有两种方法：一种是使用常量名直接获取值，另一种是使用 constant()函数，这和直接使用常量名输出的效果是一样的。但函数可以动态输出不同的常量，在使用上要灵活、方便得多。函数的语法格式为：

```
mixed constant(string const_name)
```

参数 const_name 为要获取常量的名称，也可为存储常量名的变量。如果成功则返回常量的值，失败则提示错误信息“常量没有被定义”。

要判断一个常量是否已经定义，可以使用 defined()函数。函数的语法格式为：

```
bool defined(string constant_name);
```

参数 constant_name 为要获取常量的名称，成功则返回 true，否则返回 false。

例 2.9 为了便于读者更好地理解如何定义常量，这里给出一个定义常量的实例。在这个实例中应用上述的 3 个函数：define()函数、constant()函数和 defined()函数，通过 define()函数来定义一个常量，使用 constant()函数来动态获取常量的值，应用 defined()函数来判断常量是否被定义。（**实例位置：光盘\TM\Instance\02\2.9**）

代码如下：

```
<?php
define ("MESSAGE","明日科技");                          //定义常量，并设置大小写敏感
echo MESSAGE."<BR>";                                  //输出常量 MESSAGE
echo Message."<BR>";                                  //输出"Message"，表示没有该常量
define ("COUNT","明日科技，让您尽在其中",true);
echo COUNT."<BR>";                                    //输出常量 COUNT
echo Count."<BR>";                                    //输出常量 COUNT，因为设定大小写不敏感
$name = "count";
echo constant ($name)."<BR>";                         //输出常量 COUNT
if (defined ("MESSAGE")){                             //如果定义返回 true，使用 echo 输出显示信息
    echo "明日科技是一家知名企业的软件公司！";
}
?>
```

运行结果如图 2.7 所示。

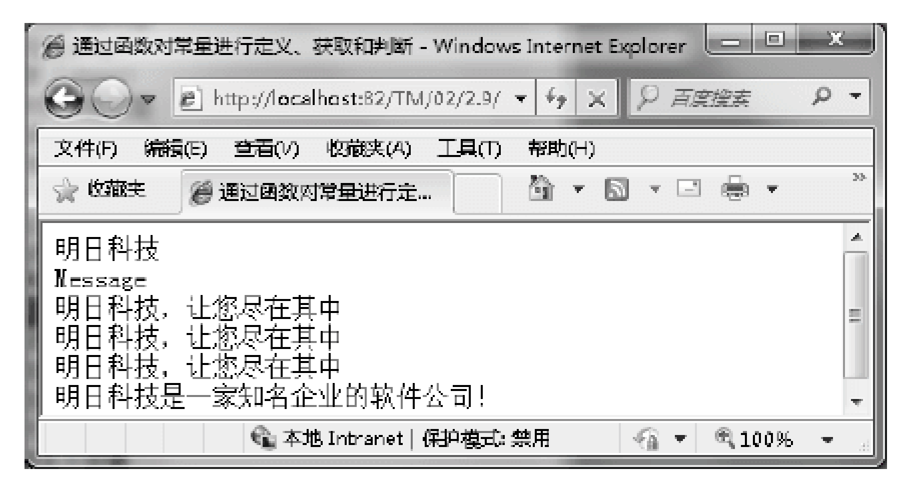


图 2.7 通过函数对常量进行定义、获取和判断

2.3.2 预定义常量

PHP 中可以使用预定义常量获取 PHP 中的信息。常用的预定义常量如表 2.7 所示。

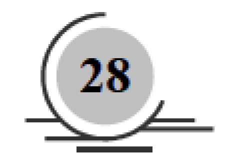


表 2.7　PHP 的预定义常量

常　量　名	功　　能
__FILE__	默认常量，PHP 程序文件名
__LINE__	默认常量，PHP 程序行数
PHP_VERSION	内建常量，PHP 程序的版本，如 3.0.8_dev
PHP_OS	内建常量，执行 PHP 解析器的操作系统名称，如 Windows
TRUE	这个常量是一个真值（true）
FALSE	这个常量是一个假值（false）
NULL	一个 null 值
E_ERROR	这个常量指到最近的错误处
E_WARNING	这个常量指到最近的警告处
E_PARSE	这个常量指解析语法有潜在问题处
E_NOTICR	这个常量为发生不寻常的提示但不一定是错误处

注意 “__FILE__”和“__LINE__”中的“__”是两条下划线，而不是一条“_”。

说明 表中以“E_”开头的预定义常量，是 PHP 的错误调试部分。如果想详细了解相关内容，可参考 error_reporting()函数。

例 2.10　预定义常量与用户自定义常量在使用上没什么差别。下面应用预定义常量来输出 PHP 中的信息。**（实例位置：光盘\TM\Instance\02\2.10）**

实例代码如下：

```
<?php
echo "当前文件路径： ".__FILE__;                          //输出“__FILE__”常量
echo "<br>当前行数：".__LINE__;                           //输出“__LINE__”常量
echo "<br>当前 PHP 版本信息：".PHP_VERSION;               //输出 PHP 版本信息
echo "<br> 当前操作系统: ".PHP_OS ;                       //输出系统信息
?>
```

运行结果如图 2.8 所示。

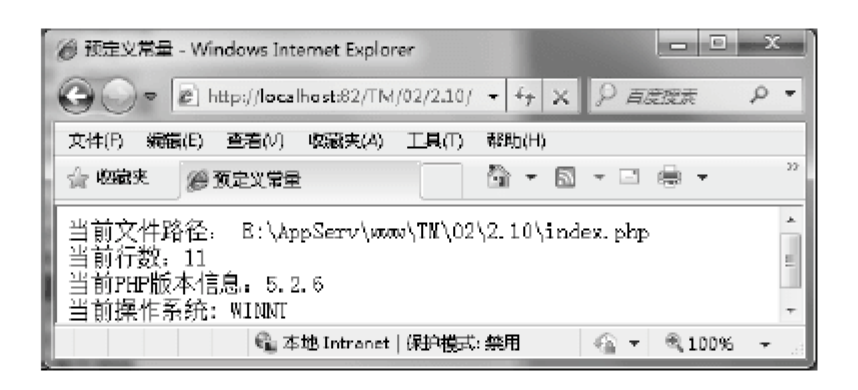


图 2.8　应用 PHP 预定义常量输出信息

注意 根据每个用户操作系统和软件版本的不同，该程序的运行结果也不一定相同。

2.4　PHP 的变量应用

视频讲解：光盘\TM\Video\第 2 章\PHP 的变量应用.exe

变量是指在程序执行过程中数字可以变化的量。变量通过一个名字（变量名）来标识。系统为程序中的每一个变量分配一个存储单元，变量名实质上就是计算机内存单元的命名。因此，借助变量名即可访问内存中的数据。

2.4.1　变量声明及使用

和很多语言不同，在 PHP 中使用变量前不需要声明变量（PHP 4.0 之前需要声明变量），只需为变量赋值即可。PHP 中的变量名称用$和标识符表示，变量名是区分大小写的。

PHP 中的变量名称遵循以下约定。

☑ 在 PHP 中的变量名是区分大小写的。

☑ 变量名必须是以美元符号（$）开始。

☑ 变量名开头可以以下划线开始。

☑ 变量名不能以数字字符开头。

☑ 变量名可以包含一些扩展字符（如重音拉丁字母），但不能包含非法扩展字符（如汉字字符和汉字字母）。

技巧 声明的变量不可以与已有的变量重名，否则将引起冲突。变量的名称应采用能反映变量含义的名称，以利于提高程序的可读性。只要能明确反映变量的含义，可以使用英文单词、单词缩写、拼音（尽量使用英文单词），如$book_name、$user_age、$shop_price 等，必要时，也可以将变量的类型包含在变量名中，如$book_id_int，这样可以直接根据变量名称了解变量的类型。

在程序中使用变量前，需要为变量赋值。PHP 中变量的定义非常简单、灵活。在定义变量时，不需要指定变量的类型，PHP 自动根据对变量的赋值决定其类型。变量的赋值是通过使用赋值运算符（=）实现的。在定义变量时也可以直接为变量赋值，此时称之为变量的初始化。

例 2.11　下面的代码定义了一个整型变量 n_sum，将其赋值为 100；定义一个布尔型变量；定义一个空字符串。**（实例位置：光盘\TM\Instance\02\2.11）**

```
<?php
	$n_sum = 100;			//定义一个整型变量，并进行初始化
	$str1=false;			//定义一个布尔型变量，并进行初始化
	$str2=" ";			//定义一个空字符串
?>
```

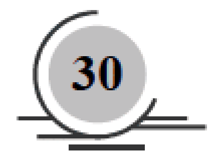
注意 在定义变量时，要养成良好的编程习惯，在定义变量前要对其定义初始值。如果在定义变量时没有指定变量的初始值，那么在使用变量时，PHP 会根据变量在语句中所处的位置确定其类型，并采用该类型的默认值。字符串的初始值为空值；整型的初始值为 0；布尔型的初始值为 false。

对变量赋值时，要遵循变量命名规则，如下面的变量命名是合法的。

```
<?php
	$helpsoft="明日科技有限公司";
	$_book="软件公司 ";
?>
```

下面的变量命名是非法的。

```
<?php
```



```
    $5_str="明日科技";                    //变量名不能以数字字符开头
    $@zts = "mrkj";                       //变量名不能以其他字符开头
?>
```

注意 PHP 中的变量名称区分大小写，而函数名称不区分大小写。

除了直接赋值外，还有两种方式来给变量声明或赋值。一种是变量间的赋值。结果为：明日科技有限公司。另一种是引用赋值。从 PHP 4.0 开始，PHP 引入了"引用赋值"的概念。引用赋值是指用不同的名字访问同一个变量内容，当改变其中一个变量的值时，另一个也跟着发生变化。使用&符号来表示引用。

例 2.12　变量间的赋值是指赋值后，两个变量使用各自的内存，互不干扰。（**实例位置：光盘\TM\Instance\02\2.12**）

实例代码如下：

```
<?php
$str1 = "明日科技有限公司";              //声明变量$str1
$str2 = $str1;                           //使用$str1 来初始化$str2
$str1 = "我喜欢学 PHP";                  //改变变量$str1 的值
echo $str2;                              //输出变量$str2 的值
?>
```

例 2.13　本例中变量$str2 是变量$str 的引用，当给变量$str 赋值后，$str2 的值也会跟着发生变化。（**实例位置：光盘\TM\Instance\02\2.13**）

实例代码如下：

```
<?php
$str = "是一家知名的软件公司";          //声明变量$str
$str2 = & $str;                          //使用引用赋值，这是$str2 已经赋值成为"是一家知名的软件公司"
$str = "明日科技：$str";                 //重新给$str 赋值
echo $str2;                              //输出变量$str2
echo "<p>";                              //输出换行标记
echo $str;                               //输出变量$str
?>
```

运行结果如图 2.9 所示。

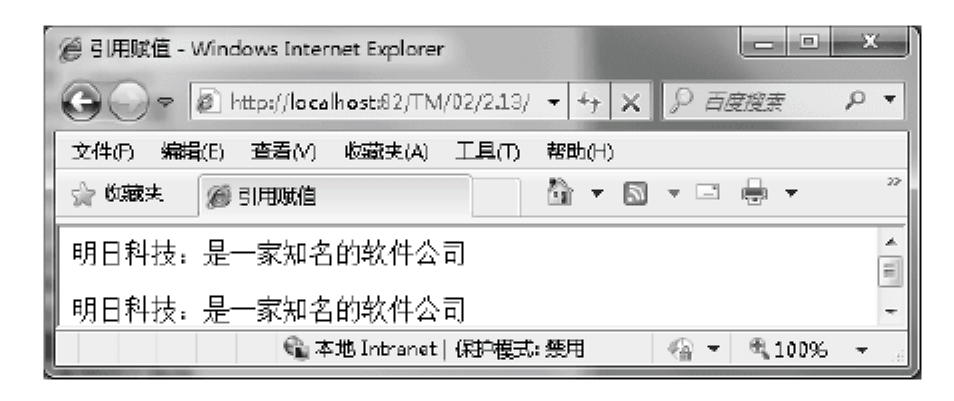


图 2.9　引用赋值

引用变量并不是复制一个变量给另一个变量，而是将两个变量指向同一个内容，可以理解为将同一个变量的地址传递给另一个变量。引用后，两个变量完全相同。当对其中的任意一个变量的内容进行更改时，另一个变量的内容也会随之更改。

引用和复制的区别在于：复制是将原变量内容复制下来，开辟一个新的内存空间来保存。而引用则是给变量的内容再命一个名字。也可以这样理解，一些论坛的版主、博客的博主，登录网站时发表帖子或文章时一般不会留真名，而是用笔名，这个笔名就可以看作是一个引用，是其身份的代表。

2.4.2　变量作用域

变量在使用时，要符合变量的定义规则。变量必须在有效范围内使用，如果变量超出有效范围，变量也就失去其意义了。按作用域可以将变量分为全局变量、局部变量和静态变量。变量作用域的说明如表 2.8 所示。

表 2.8　变量作用域

作　用　域	说　　明
全局变量	被定义在所有函数以外的变量，其作用域是整个 PHP 文件，但是在用户自定义函数内部是不可用的。想在用户自定义函数内部使用全局变量，要使用关键字 global 声明，或者通过使用全局数组$globals 进行访问
局部变量	在函数内部定义的变量，只限于在函数内部使用，在函数外部不能被使用
静态变量	能够在函数调用结束后仍保留变量值，当再次回到其作用域时，又可以继续使用原来的值。而一般变量是在函数调用结束后，其存储的数据值将被清除，所占的内存空间被释放。使用静态变量时，先要用关键字 static 来声明变量，需要把关键字 static 放在要定义的变量之前

在函数的内部定义的变量，其作用域是所在函数。如果在函数外赋值，将被认为是完全不同的另一个变量。在退出声明变量的函数时，该变量及相应的值就会被撤销。

例 2.14　下面在自定义函数中应用全局变量与局部变量进行对比。在本实例中定义两个全局变量$zy 和$zyy，在用户自定义函数 lxt()中，如果想要在第 6 行和第 8 行调用它们，而程序输出的结果是“明日科技有限公司”，因为在第 7 行用关键字 global 声明了全局变量$zyy。而第 5 行由于定义了局部变量，因此输出的结果为“明日科技”，其中第 5 行的$zy 和第 2 行的$zy 没有任何关系。**（实例位置：光盘\TM\Instance\02\2.14）**

实例代码如下：

```
<?php
$zy = "你好" ;
$zyy = "有限公司" ;
function lxt (){
    $zy="明日科技";                      //定义局部变量
    echo $zy;                            //$zy 输出的是局部变量的内容，而并非全局
    global $zyy ;                        //应用关键字 global 在函数内部定义全局变量
    echo $zyy."<br>" ;                   //此处调用$zyy
 }
 lxt () ;
?>
```

运行结果：明日科技有限公司。

注意　因为默认情况下全局变量和局部变量的作用域是不相交的，所以，在函数内部可以定义与全局变量同名的变量。全局变量可以在程序中的任何地方访问。但是在用户自定义函数内部是不可用的，想在用户自定义函数内部使用全局变量，要使用关键字 global 声明。

静态变量在函数内部定义，只局限于函数内部使用，但却具有和程序文件相同的生命周期。也就是说，静态变量一旦被定义，在当前程序文件结束之前一直存在。

静态变量通过在变量前使用关键字 static 声明变量。格式如下：

```
static $str;
```

例 2.15　下面应用静态变量和普通变量同时输出一个数据，看两者的功能有什么不同。**（实例位置：光盘\TM\Instance\02\2.15）**

实例代码如下：

```
<?php
function zdy (){
   static $message = 0 ;                 //初始化静态变量
   $message+=1;                          //静态变量加 1
   echo $message." " ;                   //输出静态变量
}
```

```
function zdy1(){
    $message = 0 ;                                  //声明函数内部变量（局部变量）
    $message += 1 ;                                 //局部变量加 1
    echo $message." " ;                             //输出局部变量
}
for ( $i=0 ; $i<10 ; $i++ )       zdy() ;           //输出 1～10
echo "<br>";
for ($i=0 ; $i<10 ; $i++ )       zdy1() ;           //输出 10 个 1
?>
```

运行结果如图 2.10 所示。

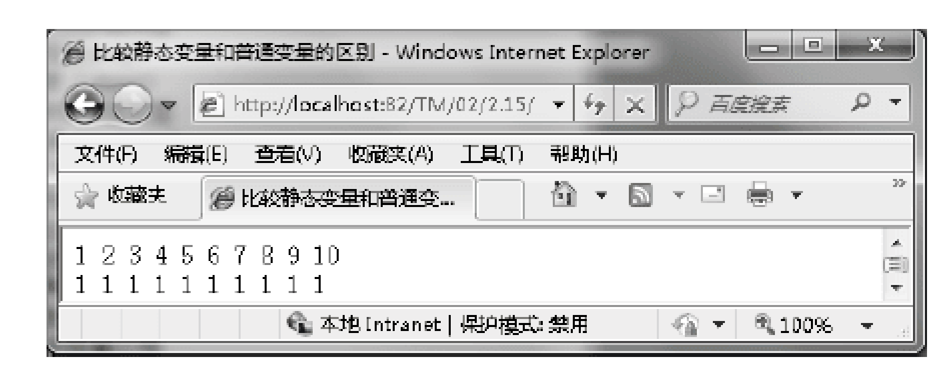


图 2.10　比较静态变量和普通变量的区别

自定义函数 zdy()是输出 1～10 的 10 个数字，而 zdy1()函数则输出的是 10 个 1。因为自定义函数 zdy()含有静态变量，而函数 zdy1()是一个普通变量。初始化都为 0，再分别使用 for 循环调用两个函数，结果是静态变量的函数 zdy()在被调用后保留了$message 中的值，静态变量的初始化只是在第一次遇到时被执行，以后就不再对其进行初始化操作了，将会略过第三行代码不执行；而普通变量的函数 zdy1()在被调用后，其变量$message 失去了原来的值，重新被初始化为 0。

2.4.3　可变变量

可变变量是一种独特的变量，变量的名称并不是预先定义好的，而是动态地设置和使用。可变变量一般是指使用一个变量的值作为另一个变量的名称，所以可变变量又称为变量的变量。可变变量通过在一个变量名称前使用两个“$”符号实现。

例 2.16　下面应用可变变量实现动态改变变量的名称。首先定义两个变量$change_name 和$php，并且输出变量$change_name 的值，然后应用可变变量来改变变量$change_name 的名称，最后输出改变名称后的变量值。**（实例位置：光盘\TM\Instance\02\2.16）**

程序代码如下：

```
<?php
$change_name = "Look";                              //声明变量$change_name
$Look = "美好的一天开始了!";                        //声明变量$php
echo $change_name ;                                 //输出变量$change_name
echo $$change_name ;                                //通过可变变量输出$php 的值
?>
```

结果为：Look 美好的一天开始了!

2.4.4　预定义变量

PHP 还提供了很多非常实用的预定义变量，通过这些预定义变量可以获取到用户会话、用户操作系统的环境和本地操作系统的环境等信息。常用的预定义变量如表 2.9 所示。

表 2.9　预定义变量

变量的名称	说　　明
$_SERVER['SERVER_ADDR']	当前运行脚本所在服务器的 IP 地址
$_SERVER['SERVER_NAME']	当前运行脚本所在服务器主机的名称。如果该脚本运行在一个虚拟主机上，则该名称是由那个虚拟主机所设置的值决定

续表

变量的名称	说　明
$_SERVER['REQUEST_METHOD']	访问页面时的请求方法，如 GET、HEAD、POST、PUT。如果请求的方式是 HEAD，则 PHP 脚本将在送出头信息后终止（这意味着在产生任何输出后，不再有输出缓冲）
$_SERVER['REMOTE_ADDR']	正在浏览当前页面用户的 IP 地址
$_SERVER['REMOTE_HOST']	正在浏览当前页面用户的主机名。反向域名解析基于该用户的 REMOTE_ADDR
$_SERVER['REMOTE_PORT']	用户连接到服务器时所使用的端口
$_SERVER['SCRIPT_FILENAME']	当前执行脚本的绝对路径名。注意，如果脚本在 CLI 中被执行，作为相对路径，如 file.php 或者../file.php，$_SERVER['SCRIPT_FILENAME']将包含用户指定的相对路径
$_SERVER['SERVER_PORT']	服务器所使用的端口，默认为 80。如果使用 SSL 安全连接，则这个值为用户设置的 HTTP 端口
$_SERVER['REMOTE_ADDR']	正在浏览当前页面用户的 IP 地址
$_SERVER['DOCUMENT_ROOT']	当前运行脚本所在的文档根目录。在服务器配置文件中定义
$_COOKIE	通过 HTTPCookie 传递到脚本的信息。这些 cookie 多数是由执行 PHP 脚本时通过 setcookie()函数设置的
$_SESSION	包含与所有会话变量有关的信息。$_SESSION 变量主要应用于会话控制和页面之间值的传递
$_POST	包含通过 POST 方法传递的参数的相关信息。主要用于获取通过 POST 方法提交的数据
$_GET	包含通过 GET 方法传递的参数的相关信息。主要用于获取通过 GET 方法提交的数据
$GLOBALS	由所有已定义的全局变量组成的数组。变量名就是该数组的索引。它可以被称为所有超级变量的超级集合

2.4.5　变量的生存周期

变量存在的时间称为生存周期，即从变量被声明的那一刻起，直到脚本运行结束。对于过程级变量，其存活期仅是该过程运行的时间，该过程结束后，变量随之消失。在执行过程中，局部变量是理想的临时存储空间。在不同过程中可以使用同名的局部变量，这是因为每个局部变量只被声明它的过程识别。

2.5　PHP 运算符

视频讲解：光盘\TM\Video\第 2 章\PHP 运算符.exe

运算符是用来对变量、常量或数据进行计算的符号，它对一个值或一组值执行一个指定的操作。PHP 的运算符包括算术运算符、字符串运算符、赋值运算符、递增或递减运算符、位运算符、逻辑运算符、比较运算符和条件运算符。下面进行详细讲解。

2.5.1　算术运算符

算术运算符用来连接运算表达式。算术运算符包括加（+）、减（-）、乘（*）、除（/）、取模（%）等运算符。常用的算术运算符如表 2.10 所示。

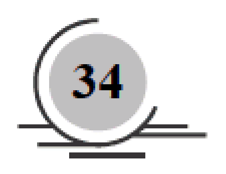


表 2.10　常用的算术运算符

算术操作符	说　　明	举例（定义$a=5，$b=3）	结　　果
+	加法运算	$a + $b	值为 8
−	减法运算	$a−$b	值为 2
*	乘法运算	$a * $b	值为 15
/	除法运算	$a / $b	值为 1.66
%	取模运算	$a % $b	值为 2

说明 在算术运算符中使用%求余，如果被除数（$a）是负数的话，那么取得的结果也是一个负值。

例 2.17　在本例中分别应用算术运算符对表达式进行计算。（**实例位置：光盘\TM\Instance\02\2.17**）

实例代码如下：

```
<?php
	$a = 6;                                         //声明变量$a
	$b = -4;                                        //声明变量$b
	$c = 10;                                        //声明变量$c
	echo "\$a = ".$a."<br>";                        //输出变量$a
	echo "\$b = ".$b."<br>";                        //输出变量$b
	echo "\$c = ".$c."<br>";                        //输出变量$c
	echo "\$a + \$b = ".($a + $b)."<br>";           //计算变量$a 加$b 的值
	echo "\$a - \$b = ".($a - $b)."<br>";;          //计算变量$a 减$b 的值
	echo "\$a * \$b = ".($a * $b)."<br>";           //计算$a 乘以$b 的值
	echo "\$a / \$b = ".($a / $b)."<br>";           //计算$a 除以$b 的值
	echo "\$a % \$c = ".($a % $c);                  //计算$a 和$c 的余数，被除数为 15
?>
```

运行结果如图 2.11 所示。

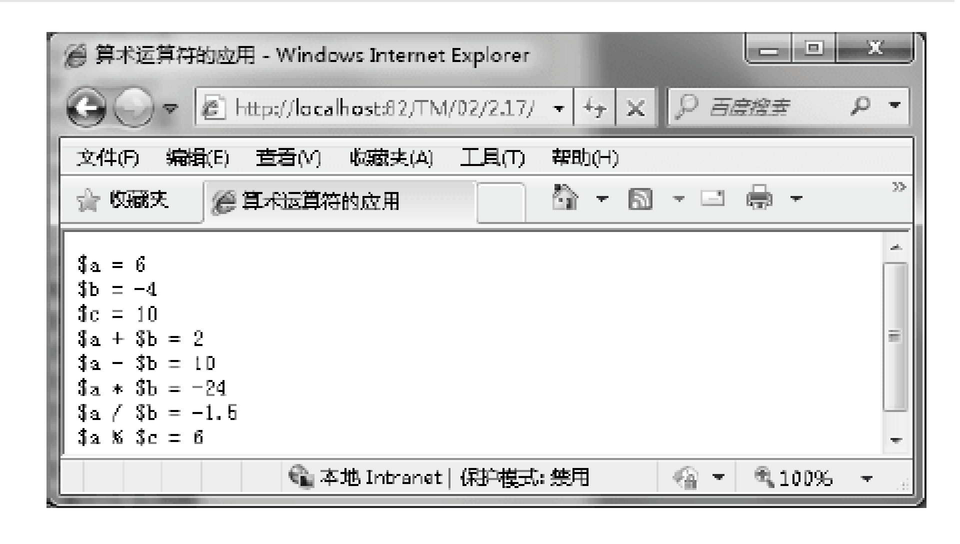


图 2.11　算术运算符的应用

2.5.2　字符串运算符

字符串运算符只有一个，即英文的句号“.”。它将两个字符串连接起来，形成一个新的字符串。使用过 C 或 Java 的读者要注意了，这里的“+”号，只做赋值运算符使用，而不能做字符串运算符。

例 2.18　下面应用一个实例，来看一下（.）和“+”两者之间的区别。当使用“.”时，变量$m 和$n 两个字符串组成一个新的字符“3.5e11”，当使用“+”时，PHP 会认为这是一次运算。如果“+”号的两边有字符类型，则自动转换为整型；如果是字母，则输出为 0；如果是以数字开头的字符串，则会截取字串头部的数字，再进行运算。（**实例位置：光盘\TM\Instance\02\2.18**）

实例代码如下：

```
<?php
$n = "3.5e";                                    //声明一个字符串变量，以数字开头
$m =11;                                         //声明一个整型变量
$nm = $n.$m;                                    //使用“.”运算符将两个变量连接
echo $nm."\t";
$mn = $n + $m ;                                 //使用“+”运算符将两个变量连接
echo $mn;
?>
```

结果为：3.5e11　14.5。

2.5.3　赋值运算符

最基本的赋值运算符是“=”，用于对变量进行赋值，而其他运算符可以和赋值运算符“=”联合使用，构成组合赋值运算符。赋值运算符是把基本赋值运算符“=”右边的值赋给左边的变量或者常量。在 PHP 中的赋值运算符如表 2.11 所示。

表 2.11　常用赋值运算符

操　作	符　号	举　例	展开形式	意　义
赋值	=	$a=b	$a=b	将右边表达式的值赋给左边的变量
加	+=	$a+= b	$a=$a + b	将运算符左边的变量加上右边表达式的值赋给左边的变量
减	−=	$a−= b	$a=$a−b	将运算符左边的变量减去右边表达式的值赋给左边的变量
乘	*=	$a*= b	$a=$a * b	将运算符左边的变量乘以右边表达式的值赋给左边的变量
除	/=	$a/= b	$a=$a / b	将运算符左边的变量除以右边表达式的值赋给左边的变量
连接字符	.=	$a.= b	$a=$a. b	将右边的字符加到左边
取余数	%=	$a%= b	$a=$a % b	将运算符左边的变量用右边表达式的值求模，并将结果赋给左边的变量

例 2.19　应用赋值运算符给指定的变量赋值，并计算各表达式的值。（**实例位置：光盘\TM\Instance\02\2.19**）

代码如下：

```
<?php
	$a = 8;                                          //声明变量$a
	$b =4;                                           //声明变量$b
	echo "\$a = ".$a."<br>";                         //输出变量$a
	echo "\$b = ".$b."<br>";                         //输出变量$b
	echo "\$a += \$b = ".($a += $b)."<br>";          //计算变量$a 加$b 的值
	echo "\$a -= \$b = ".($a -= $b)."<br>";          //计算变量$a 减$b 的值
	echo "\$a *= \$b = ".($a *= $b)."<br>";          //计算$a 乘以$b 的值
	echo "\$a /= \$b = ".($a /= $b)."<br>";          //计算$a 除以$b 的值
	echo "\$a %= \$b = ".($a %= $b);                 //计算$a 和$b 的余数
?>
```

运行结果如图 2.12 所示。

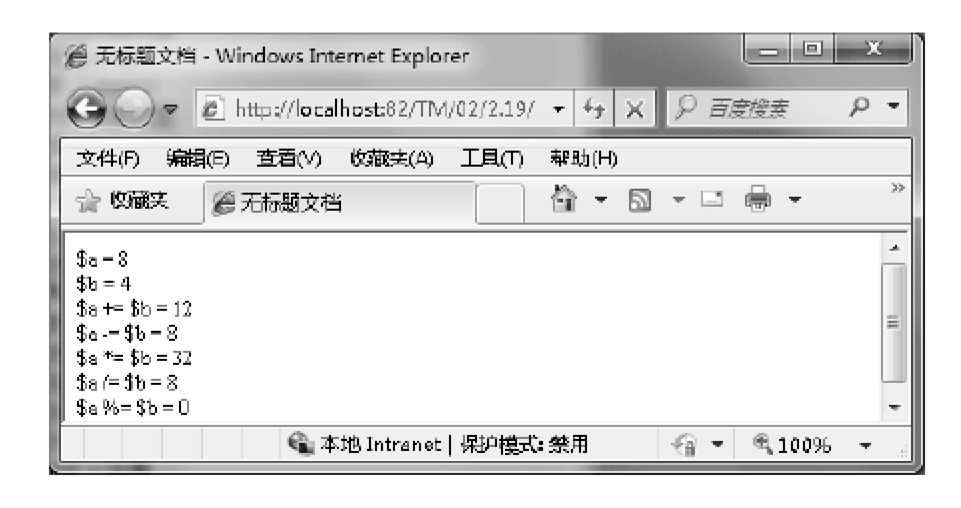


图 2.12　赋值运算符的应用

2.5.4　递增或递减运算符

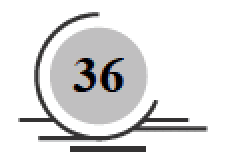
算术运算符适合在有两个或者两个以上不同操作数的场合使用，但是，当只有一个操作数时，使用算术运算符是没有必要的。这时，就可以使用“++”或者“--”运算符了，即递增或递减运算符。

递增或递减运算符有两种使用方法，一种是先将变量增加或者减少 1 后再将值赋给原变量，称为前置递增或递减运算符（也称前置自增自减运算符）；另一种是将运算符放在变量后面，即先返回变量的当前值，然后变量的当前值增加或者减少 1，称为后置递增或递减运算符（后置自增自减运算符）。

1．前置递增运算符

前置递增运算符是指使用前置“++”运算符将变量加 1 再将值赋给原变量。例如：

```
<?php
```



```
        $a=5;
        echo ++$a;
?>
```

结果为：6。

该程序的运行结果为 6，也就是首先把变量$a 加 1，再将结果赋给原变量，所以运行本程序后，原变量$a 的值已经变成 6。

2．前置递减运算符

前置递减运算符是指使用前置“--”运算符将变量减 1 再将值赋给原变量。例如：

```
<?php
        $a=5;
        echo --$a;
?>
```

结果为：4。

该程序的运行结果为 4，也就是首先把变量$a 减 1，再将结果赋给原变量，所以运行本程序后，原变量$a 的值已经变成 4。

3．后置递增运算符

后置递增运算符是指先返回变量的当前值，然后再使用后置“++”运算符将变量的当前值再加 1。例如：

```
<?php
        $a=5;
        echo   $a++;
?>
```

结果为：5。

该程序的运行结果为 5，即先将变量的原值输出，然后再加 1，虽然显示输出的是 5，但实际上变量$a 的值已经是 6 了。

4．后置递减运算符

后置递减运算符是指先返回变量的当前值，然后使用后置“--”运算符将变量的当前值再减 1。例如：

```
<?php
        $a=5;
        echo   $a--;
?>
```

结果为：5。

该程序的运行结果为 5，即先将变量的原值输出，然后再减 1，虽然显示输出的是 5，但实际上变量$a 的值已经是 4 了。

2.5.5 位运算符

位运算符是指对二进制位从低位到高位对齐后进行运算。在 PHP 中的位运算符如表 2.12 所示。

表 2.12 位运算符

运 算 符	说 明	举 例
&	按位与	$m & $n
|	按位或	$m | $n
^	按位异或	$m ^ $n
~	按位取反	$m ~ $n
<<	向左移位	$m << $n
>>	向右移位	$m >> $n

例 2.20 应用位运算符对变量中的值进行位运算操作。（**实例位置：光盘\TM\Instance\02\2.20**）

实例代码如下：

```
<?php
$m =7 ;
$n =8 ;
$mn = $m & $n ;                    //位与
echo $mn ." \t";
$mn = $m | $n ;                    //位或
echo $mn ." \t";
$mn = $m ^ $n ;                    //位异或
echo $mn ." \t";
$mn = ～$m ;                       //位取反
echo $mn ;
?>
```

结果为：0　15　15　−8。

2.5.6 逻辑运算符

逻辑运算符用来组合逻辑运算的结果，是程序设计中一组非常重要的运算符。PHP 的逻辑运算符如表 2.13 所示。

表 2.13　PHP 的逻辑运算符

逻辑运算符	说　　明	结 果 为 真
&&或 and	逻辑与，只有当两个操作数的值都为 true 时，$m && $n 的值才为 true	当$m 和$n 都为真时
||或 or	逻辑或，只要两个操作数中有一个值为 true，$m || $n 的值就为 true	当$m 为真或者$n 为真时
xor	逻辑异或，当两个操作数的值为一真一假时，$m xor $n 的值为 true	当$m、$n 一真一假时
!	逻辑非，如果其单一操作数为 true，则返回 false；否则返回 true	当$m 为假时

在逻辑运算符中，逻辑与和逻辑或这两个运算符有 4 种运算符号（&&、and、||和 or），其中属于同一个逻辑结构的两个运算符号（如&&和 and）之间却有着不同的优先级。

例 2.21 下面分别应用逻辑或中的运算符号“||”和“or”进行相同的判断，但是因为同一逻辑结构的两个运算符（“||”和“or”）的优先级不同，输出的结果也不同。（**实例位置：光盘\TM\Instance\02\2.21**）

实例代码如下：

```
<?php
    $i = true;                    //声明一个布尔变量$i，赋值为真
    $j = true;                    //再声明一个布尔变量$j，赋值也为真
    $z = false;                   //最后再声明一个初值为假的布尔变量$z
    if($i or $j and $z)           //这是用 or 做判断
        echo "true";              //如果 if 表达式为真，输出 true
    else
        echo "false";             //否则输出 false
        echo "<br>";
    if($i || $j and $z)           //这是用||做判断
        echo "true";              //如果表达式为真，输出 true
    else
        echo "false";             //如果表达式为假，输出 false
?>
```

结果为：true

false。

注意 可以看到，两个 if 语句，除了 or 和||不同之外，其他完全一样，但最后的结果却是正好相反。在实际应用中要多注意这样的细节。

2.5.7　比较运算符

比较运算符就是对变量或表达式的结果进行大小、真假等比较，如果比较结果为真，则返回 true；如果为假，则返回 false。PHP 中的比较运算符如表 2.14 所示。

表 2.14　PHP 的比较运算符

运　算　符	说　　明	举例（定义$m=5，$n=2）	结　　果
<	小于	$m<$n	false
>	大于	$m>$n	true
<=	小于等于	$m<=$n	false
>=	大于等于	$m>=$n	true
= =	相等	$m= =$n	false
!=	不等	$m!=$n	true
= = =	恒等	$m= = = $n	false
!==	非恒等	$m!==$n	true

比较运算“= = =”和“!= =”不太常见。“$m = = = $n”，说明$m 和$n 不只是数值上相等，而且两者的类型也一样。“!= =”和“= = =”的含义差不多，“$m != = $n”是指$m 和$n 或者数值不相等，或者类型不相等。

例 2.22　下面应用比较运算符对变量中的值进行比较，设置变量$value ="100"，变量的类型为字符串型，将变量$value 与数字 100 进行比较，会发现比较的结果非常有趣。其中使用的 var_dump()函数是系统函数，作用是输出变量的相关信息。**（实例位置：光盘\TM\Instance\02\2.22）**

实例代码如下：

```
<?php
$value="100";                          //声明一个字符串变量$value
echo "\$value = \"$value\"";
echo "<p>\$value==100: ";
var_dump($value==100);                 //结果为：bool(true)
echo "<p>\$value==true: ";
var_dump($value==true);                //结果为：bool(true)
echo "<p>\$value!=null: ";
var_dump($value!=null);                //结果为：bool(true)
echo "<p>\$value==false: ";
var_dump($value==false);               //结果为：bool(false)
echo "<p>\$value === 100: ";
var_dump($value===100);                //结果为：bool(false)
echo "<p>\$value===true: ";
var_dump($value===true);               //结果为：bool(false)
echo "<p>(10/2.0 !== 5): ";
var_dump(10/2.0 !==5);                 //结果为：bool(true)
?>
```

运行结果如图 2.13 所示。

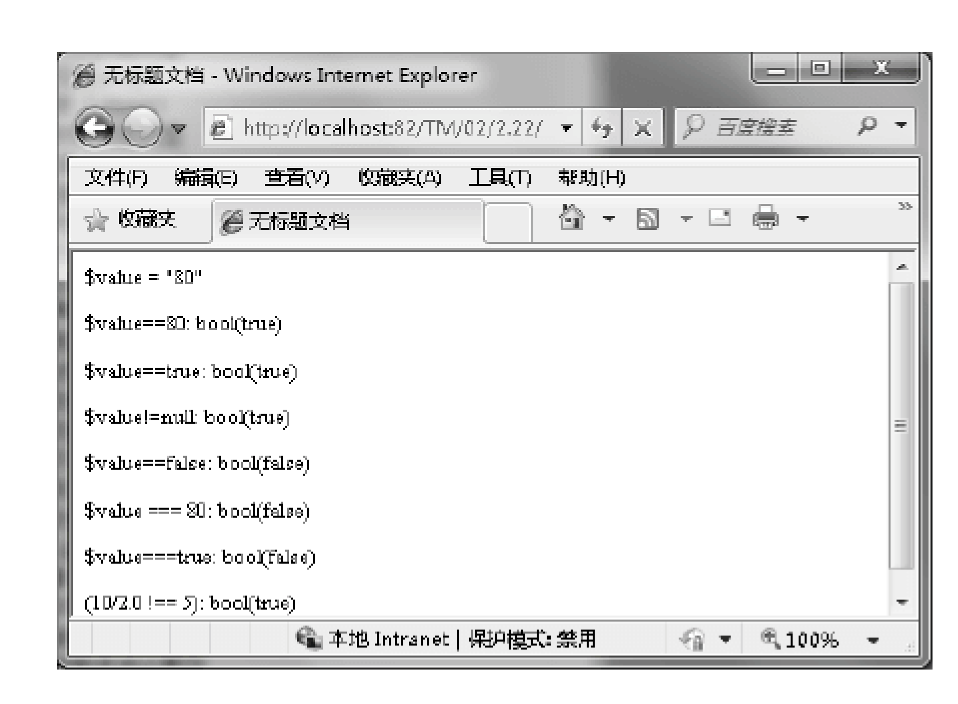


图 2.13　比较运算符的应用

2.5.8　条件运算符

条件运算符用于根据一个表达式在另两个表达式中选择一个，而不是用来在两个语句或者程序中选择。条件运算符也称三元运算符，条件运算符的使用最好放在括号里。

例 2.23　下面应用条件运算符实现一个简单的判断功能，如果变量$age 的值大于等于 9，则输出“小红是好人”，否则输出“小红是坏人”。**（实例位置：光盘\TM\Instance\02\2.23）**

实例代码如下：

```
<?php
$age=12;
echo ($age>=9)?小红是好人: 小红是坏人;
?>
```

结果为：小红是好人。

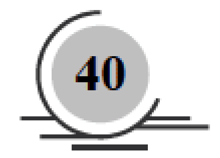
2.5.9　运算符的优先顺序和结合规则

所谓运算符的优先级，是指在表达式中哪一个运算符先计算，哪一个后计算，与数学的四则运算遵循的“先乘除，后加减”是一个道理。

PHP 的运算符在运算中遵循的规则是：优先级高的操作先执行，优先级低的操作后执行，同一优先级的操作按照从左到右的顺序进行。也可以像四则运算那样使用小括号，括号内的运算最先进行。PHP 运算符优先级的一览表如表 2.15 所示。

表 2.15　运算符的优先级

优 先 级 别	运　算　符
1	or、and、xor
2	=、+=、-=、*=、/=、.=、%=
3	||、&&
4	|、^
5	&、.
6	++、--（递增或递减运算符）
7	/、*、%
8	<<、>>
9	++、--
10	+、-（正、负号运算符）、!、~
11	==、!=、<>
12	<、<=、>、>=
13	?:
14	->
15	=>



2.6　PHP 函数

视频讲解：光盘\TM\Video\第 2 章\PHP 函数.exe

在开发过程中，经常要重复某种操作或处理，如数据查询、字符操作等，如果每个模块的操作都要重新输入一次代码，不仅令程序员头痛不已，而且对于代码的后期维护及运行效果也有着较大的影响，使用 PHP 函数即可让这些问题迎刃而解。下面将介绍这些知识。

2.6.1　定义和调用函数

函数，就是将一些重复使用到的功能写在一个独立的代码块中，在需要时单独调用。创建函数的基本语法格式为：

```
function fun_name($str1,$stgr2…$strn){
    fun_body;
}
```

其中，

☑　function：为声明自定义函数时必须使用到的关键字。

☑　fun_name：为自定义函数的名称。

☑　$str1…$strn：为函数的参数。

☑　fun_body：为自定义函数的主体，是功能实现部分。

当函数被定义好后，所要做的就是调用这个函数。调用函数地操作十分简单，只需要引用函数名并赋予正确的参数即可完成函数的调用。

例 2.24　在本例中定义了一个函数 example()，计算传入参数的平方，然后连同表达式和结果全部输出。**（实例位置：光盘\TM\Instance\02\2.24）**

实例代码如下：

```
<?php
    /*    声明自定义函数      */
    function example($num){
        return "$num * $num = ".$num * $num;          //返回计算后的结果
    }
    echo example(5);                                  //调用函数
?>
```

结果为：5 * 5 = 25。

2.6.2　在函数间传递参数

在调用函数时，需要向函数传递参数，被传入的参数称为实参，而函数定义的参数为形参。参数传递的方式有按值传递、按引用传递和默认参数 3 种。

1．按值传递方式

将实参的值复制到对应的形参中，在函数内部的操作针对形参进行，操作的结果不会影响到实参，即函数返回后，实参的值不会改变。

例 2.25　本例首先定义一个函数 example()，功能是将传入的参数值做一些运算后再输出。接着在函数

外部定义一个变量$m，也就是要传进来的参数。最后调用函数 example($m)，输出函数的返回值$m 和变量$m 的值。（**实例位置：光盘\TM\Instance\02\2.25**）

实例代码如下：

```
<?php
function example( $m ){                    //定义一个函数
    $m = $m * 4 + 8;
echo "在函数内：\$m = ".$m;                 //输出形参的值
}
$m = 1;
example( $m ) ;                            //传递值，将$m 的值传递给形参$m
echo "<p>在函数外  \$m = $m <p>" ;          //实参的值没有发生变化，输出 m=1
?>
```

运行结果如图 2.14 所示。

2．按引用传递方式

按引用传递就是将实参的内存地址传递到形参中。这时，在函数内部的所有操作都会影响到实参的值，返回后，实参的值会发生变化。引用传递方式就是传值时在原基础上加&号即可。

例 2.26　仍然使用例 2.25 中的代码，唯一不同的地方就是多了一个&号。（**实例位置：光盘\TM\Instance\02\2.26**）

实例代码如下：

```
<?php
function example( &$m ){                   //定义一个函数，同时传递参数$m 的变量
    $m = $m * 4 + 8;
echo "在函数内：\$m = ".$m;                 //输出形参的值
}
$m = 1;
example( $m ) ;                            //传递值：将$m 的值传递给形参$m
echo "<p>在函数外：\$m = $m <p>" ;          //实参的值发生变化，输出 m=15
?>
```

运行结果如图 2.15 所示。

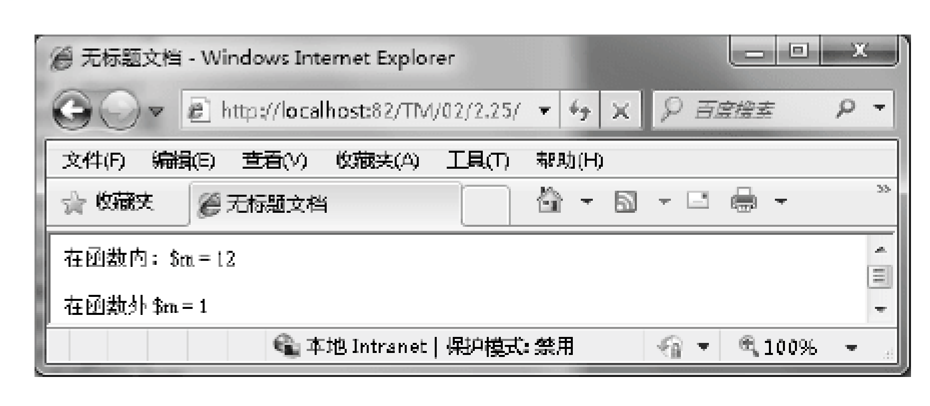


图 2.14　接值传递方式



图 2.15　按引用传递方式

3．默认参数（可选参数）

还有一种设置参数的方式，即可选参数。可以指定某个参数为可选参数，将可选参数放在参数列表末尾，并且指定其默认值为空。

例 2.27　本例使用可选参数实现一个简单的价格计算功能，设置自定义函数 values 的参数 $tax 为可选参数，其默认值为空。第一次调用该函数，并且给参数$tax 赋值 0.36，输出价格；第二次调用该函数，不给参数$tax 赋值，输出价格。（**实例位置：光盘\TM\Instance\02\2.27**）

实例代码如下：

```
<?php
    function values($price,$tax=""){          //定义一个函数，其中的一个参数初始值为空
        $price=$price+($price*$tax);          //声明一个变量$price，等于两个参数的运算结果
```



```
        echo "价格:$price<br>";                    //输出价格
    }
    values(100,0.36);                               //为可选参数赋值 0.36
    values(100);                                    //没有给可选参数赋值
?>
```

结果为：价格:136　　价格:100。


当使用默认参数时，默认参数必须放在非默认参数的右侧，否则函数可能出错。


从 PHP 5 开始，默认值也可以通过引用传递。

2.6.3　从函数中返回值

在 2.6.1 节中介绍了如何定义和调用一个函数，并且讲解了如何在函数间传递值，本节将讲解函数的返回值。通常，函数将返回值传递给调用者的方式是使用关键字 return。

return 将函数的值返回给函数的调用者，即将程序控制权返回到调用者的作用域。如果在全局作用域内使用 return 关键字，那么将终止脚本的执行。

例 2.28　本实例使用 return()函数返回一个操作数。先定义函数 values()，函数的作用是输入物品的单价、重量，然后计算总金额，最后输出商品的价格。**（实例位置：光盘\TM\Instance\02\2.28）**

实例代码如下：

```
<?php
    function values($price,$tax=0.25){              //定义一个函数，函数中的一个参数有默认值
        $price=$price+($price*$tax);                //计算物品金额
        return $price;                              //返回金额
    }
    echo values(100);                               //调用函数
?>
```

结果为：125。

return 语句只能返回一个参数，也只能返回一个值，不能一次返回多个。如果要返回多个结果，就要在函数中定义一个数组，将返回值存储在数组中返回。

2.6.4　变量函数

PHP 支持变量函数。下面通过一个实例来介绍变量函数的具体应用。

例 2.29　本例首先定义了 3 个函数，接着声明一个变量，通过变量来访问不同的函数。**（实例位置：光盘\TM\Instance\02\2.29）**

实例代码如下：

```
<?php
function come() {                                   //定义 come()函数
    echo "来了<p>";
}
function go($name = "jack") {                       //定义 go()函数
    echo " $name. '走了<p>";
```

```
}
function back($string)                        //定义 back()函数
{
    echo "又回来了，$string<p>";
}
$func = "come";                               //声明一个变量，将变量赋值为"come"
$func();                                      //使用变量函数来调用函数 come()
$func = "go";                                 //重新给变量赋值
$func("Tom");                                 //使用变量函数来调用函数 go()
$func = "back";                               //重新给变量赋值
$func("Lily");                                //使用变量函数来调用函数 back()
?>
```

运行结果如图 2.16 所示。

可以看到，函数的调用是通过改变变量名来实现的，通过在变量名后面加上一对小括号，PHP 将自动寻找与变量名相同的函数，并且执行它。如果找不到对应的函数，系统将会报错。这个技术可以用于实现回调函数和函数表等。

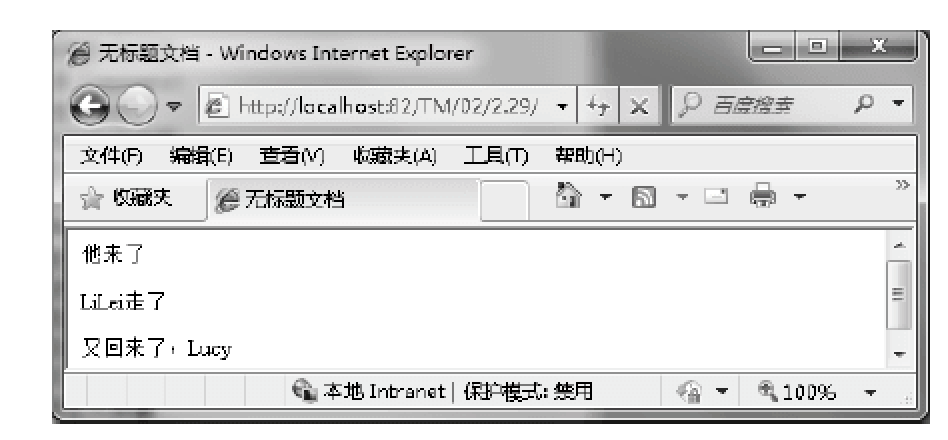


图 2.16　变量函数

2.6.5　对函数的引用

在 2.6.2 节的参数传递中有按引用传递的方式，可以修改实参的内容。引用不仅用于普通变量、函数参数，也可作用于函数本身。对函数的引用，就是对函数返回结果的引用。

例 2.30　在本例中，首先定义一个函数，这里需在函数名前加"&"号，接着，变量$str 将引用该函数，最后输出该变量$str，实际上就是$tmp 的值。（**实例位置：光盘\TM\Instance\02\2.30**）

实例代码如下：

```
<?php
function &example($tmp=0){                    //定义一个函数，加"&"号
    return $tmp;                              //返回参数$tmp
}
$str = &example("明日科技");                  //声明一个函数的引用$str
echo $str."<p>";                              //输出$str
?>
```

结果为：明日科技。



注意　和参数传递不同，这里必须在两个地方使用"&"号，用来说明返回的是一个引用。

2.6.6　取消引用

当不再需要引用时，可以取消引用。取消引用使用 unset()函数，它只是断开了变量名和变量内容之间的绑定，而不是销毁变量内容。

例 2.31　本例首先声明一个变量和变量的引用，输出引用后取消引用，再次调用引用和原变量。可以看到，取消引用后对原变量没有任何影响。（**实例位置：光盘\TM\Instance\02\2.31**）

实例代码如下：

```
<?php
    $num = 789;                               //声明一个整型变量
    $math = &$num;                            //声明一个对变量$num 的引用$math
```



```
    echo "\$math is:  ".$math."<br>";           //输出引用$math
    unset($math);                               //取消引用$math
    echo "\$math is: ".$math."<br>";            //再次输出引用
    echo "\$num is:   ".$num;                   //输出原变量
?>
```

运行结果如图 2.17 所示。

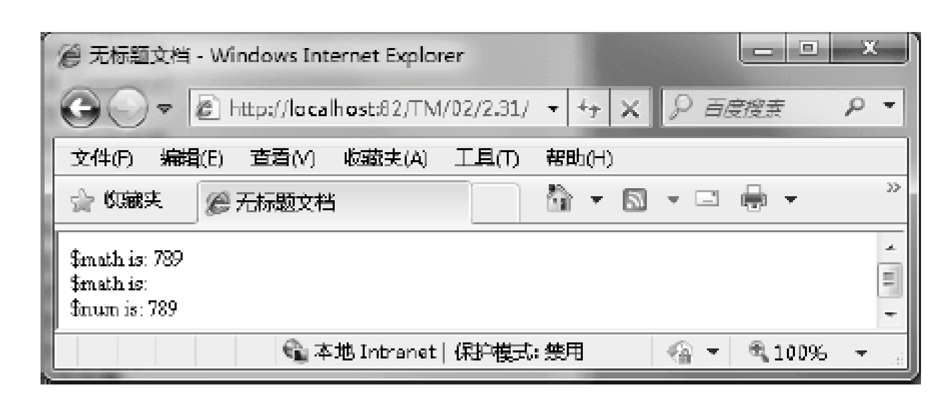


图 2.17 取消引用

2.7 输 出 语 句

视频讲解：光盘\TM\Video\第 2 章\输出语句.exe

程序设计中经常需要将字符串或者字符串变量输出到浏览器。本节将介绍几种输出语句的方法。由于 PHP 中数据类型是可以自动转换的，所以这些方法也可用于输出其他类型的数据。

2.7.1 应用 print 语句输出字符

print 输出语句的作用是将字符串输出到浏览器或打印机等输出设备，通常输出到浏览器。print 语句只可以同时输出一个字符串，需要圆括号。echo 语句可以同时输出多个字符串，并不需要圆括号。print 语句和 echo 语句的功能是相同的，但 print 语句有返回值，而 echo 语句没有返回值，所以 echo 语句相对可能快些，但这种速度差异并不明显。

语法：

```
int print (string arg)
```

该语句执行成功则返回 1，失败则返回 0。



注意 使用 print 语句时，括号可以省略。

例 2.32 应用 print 语句输出字符串“PHP 编程词典，让您编程无忧”。（**实例位置：光盘\TM\Instance\02\2.32**）

代码如下：

```
<?php
print ("PHP 编程词典，让您编程无忧");
?>
```

运行结果如图 2.18 所示。

例 2.33 上面讲解了应用 print 语句输出字符串的方法，接下来应用 print 语句输出变量中的字符串。（**实例位置：光盘\TM\Instance\02\2.33**）

代码如下：

```
<?php
$str="明日图书辉煌打造";
```

```
print $str;
?>
```

运行结果如图 2.19 所示。



图 2.18　应用 print 语句输出字符串



图 2.19　应用 print 语句输出变量中的字符串

例 2.34　应用 print 语句输出新型字符串中的值。（**实例位置：光盘\TM\Instance\02\2.34**）

代码如下：

```
<?php
print <<<END
吉林省明日科技有限公司
END;
?>
```

运行结果如图 2.20 所示。

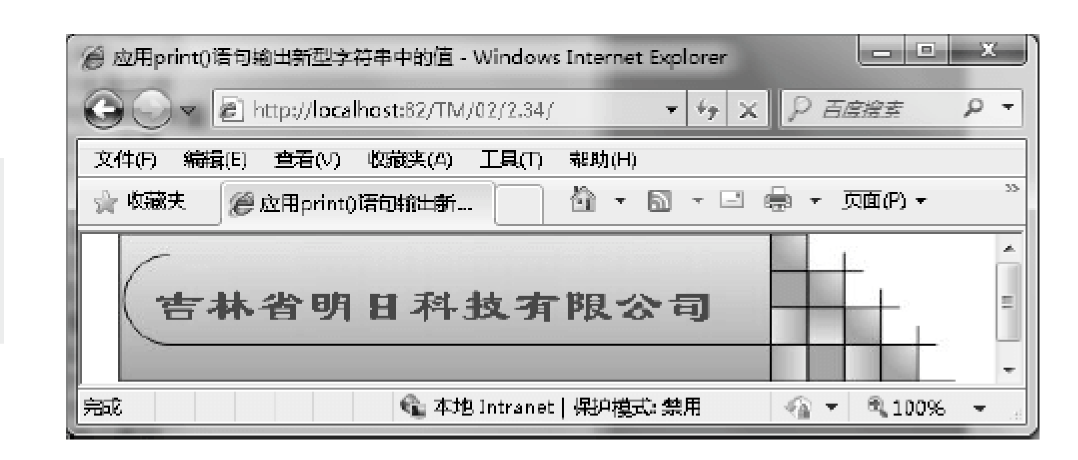


图 2.20　应用 print 语句输出新型字符串中的值

注意　由于这是一个语言结构，而非函数，因此它无法被变量函数调用。

2.7.2　应用 echo 语句输出字符

echo 语句用于在脚本当前位置输出字符串，该语句在前面章节所举的实例中多次使用过，是程序设计中最常使用的一种输出方法。echo 使用方式非常简单，直接将要输出的字符串或者变量跟在 echo 后面，字符串或变量和 echo 之间以一个空格分隔。

语法：

```
echo "string arg1, string [arg2]..."
```

echo 语句会将传入的字符串参数（arg1）进行输出。由于它本身并不是一个真正的函数，因此没有返回值。

使用 echo 语句执行的结果是，将传递给其自身的字符串输出到浏览器。下面应用 echo 语句输出字符串“明日科技图书网：www.mingribook.com”到浏览器。代码如下：

```
<?php
echo "明日科技图书网：www.mingribook.com" ;
?>
```

结果为，明日科技图书网：www.mingribook.com。

接下来应用 echo 语句输出变量，代码如下。

```
<?php
$string = "明日编程词典网：www.mrbccd.com";
echo $string;
?>
```

结果为，明日编程词典网：www.mrbccd.com。

echo 语句后面可以连续输出多个字符串，它们之间以逗号分隔。例如：

```
<?php
$mr="明日科技";
$str="是一家以计算机软件技术为核心的高科技型企业";
echo $mr,$str;
?>
```

结果为：明日科技是一家以计算机软件技术为核心的高科技型企业。

连续输出多个字符串，可以使用字符串运算符“.”将多个字符串连接成一个新型字符串输出。例如：

```
<?php
$str1="一个人的快乐";
$str2="不是因为他拥有的多";
$str3="而是因为他计较的少。";
echo $str1.",".$str2.",".$str3."。";
?>
```

结果为：一个人的快乐，不是因为他拥有的多，而是因为他计较的少。

echo 语句后紧跟的参数应该是字符串类型，如果不是，PHP 将自动将其进行转换，所以 echo 语句同样可以输出其他类型的数据。例如：

```
<?php
$name="木讷";
$age="25";
$payfor="2400.00";
echo $name." 年龄：".$age." 工资：".$payfor;
?>
```

结果为：木讷 年龄：25 工资：2400.00。

例 2.35　应用 echo 语句直接输出 phpinfo()函数调用的内容。**（实例位置：光盘\TM\Instance\02\2.35）**

代码如下：

```
<?php
echo phpinfo();
?>
```

运行结果如图 2.21 所示。

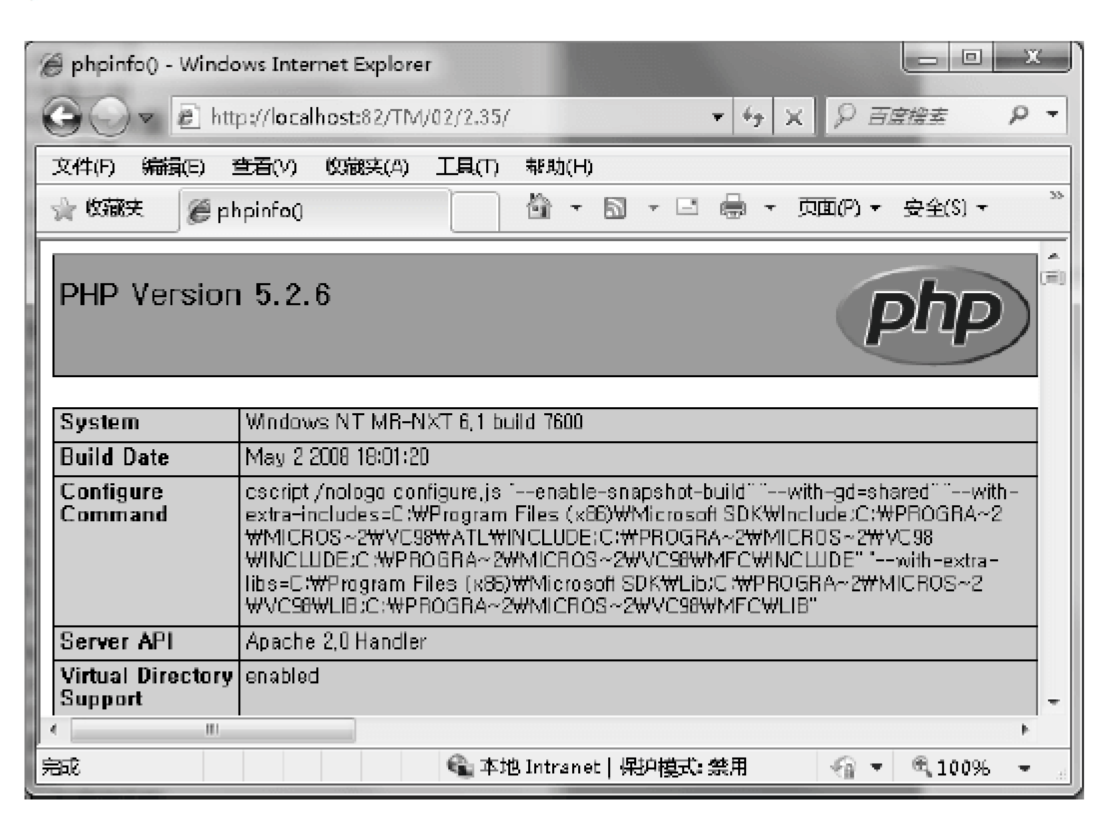


图 2.21　应用 echo 语句直接输出函数中的内容

2.7.3　应用 printf 语句格式化输出字符

printf 语句将字符串按照某种格式进行输出。

语法：

```
int printf ( string format [, mixed args [, mixed ...]] )
```

printf 语句按照参数 format 指定的内容格式对字符串进行格式化，具体参数 format 的转换格式是以百分比符号（%）开始到转换字符为止。参数 format 的格式转换类型如表 2.16 所示。

表 2.16　参数 format 的格式转换类型

参　数	说　明
b	整数转换成二进制
c	整数转换成对应的 ASCII 字符
d	整数转换成十进制
e	将整数转换成以科学计数法显示
f	被精确的数字转换成浮点数
o	整数转换成八进制
s	整数转换成字符串
u	整数转换成无符号整数
x	整数转换成小写十六进制
X	整数转换成大写十六进制

例 2.36　应用 printf 语句将整数转换成不同的类型，并以格式化的方式输出字符串。**（实例位置：光盘\TM\Instance\02\2.36）**

代码如下：

```
<?php
$string = 15;
printf("%c",$string);                    //将整数转换成 ASCII 字符
$string = "明日科技";
printf($string);                         //直接输出字符
$string = 1234.00;
printf("%0.2f",$string);                 //将整数转换成小数点后有两位小数的浮点数
printf("%s",$string);                    //将整数转换成字符串
printf("%X",$string);                    //将整数转换成大写十六进制
?>
```

运行结果为：☼明日科技 1234.0012344D2。

2.7.4　应用 sprintf 语句格式化输出字符

格式化字符串用于按一定的格式输出含有许多变量的文本，是最常用的一种操作，可以使用 PHP 的 sprintf 语句来完成格式化字符串的功能。

语法：

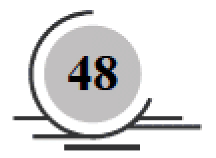
```
string sprintf(string format, mixed args...)
```

参数 format 为转换后的格式，各个变量都以“%”后的字符规定其格式，格式转换类型如表 2.16 所示。

例 2.37　应用 sprintf 语句格式化字符串。**（实例位置：光盘\TM\Instance\02\2.37）**

代码如下：

```
<?php
$name= "木讷";
$pay= 2400;
$expend= 154.5;
$balance= $pay-$expend;
echo sprintf("%s:您的本月工资为￥%0.1f 元",$name,$balance)
?>
```



运行结果为：木讷:您的本月工资为￥2245.5 元。

2.8 引 用 文 件

视频讲解：光盘\TM\Video\第 2 章\引用文件.exe

引用文件是指将另一个源文件的全部内容包含到当前源文件中进行使用。引用外部文件可以减少代码的重用性，是 PHP 编程的重要技巧。PHP 提供了 include 语句、require 语句、include_once 语句和 require_once 语句用于实现引用文件。这 4 种语句在使用上有一定的区别。下面分别进行详细讲解。

2.8.1 应用 include 语句引用文件

使用 include 语句引用外部文件时，只有代码执行到 include 语句时才将外部文件引用进来并读取文件的内容，当所引用的外部文件发生错误时，系统只给出一个警告，而整个 PHP 文件则继续向下执行。下面介绍 include 语句的使用方法。

语法：

```
void include(string filename);
```

参数 filename 是指定的完整路径文件名。



include 语句必须放到 PHP 标记中，否则代码会被视为文本而不会被执行。

例 2.38 如果 Web 页面具有一致的外观，可以在 PHP 中使用 include 语句将模板和标准元素载入到页面中。下面讲解应用 include 语句引用外部文件的方法。**（实例位置：光盘\TM\Instance\02\2.38）**

（1）首先创建一个简单的 PHP 动态页，命名为 index.php。

（2）然后应用 include 语句嵌入 3 个外部文件，分别是 top.php 页、main.php 页和 bottom.php 页。代码如下：

```
<table width="975" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td><?php include("top.php");?></td>
  </tr>
  <tr>
    <td><?php include("main.php");?></td>
  </tr>
  <tr>
    <td><?php include("bottom.php");?></td>
  </tr>
</table>
```

（3）最后在以上 3 个外部文件中分别调用图片资源及数据信息。运行结果如图 2.22 所示。

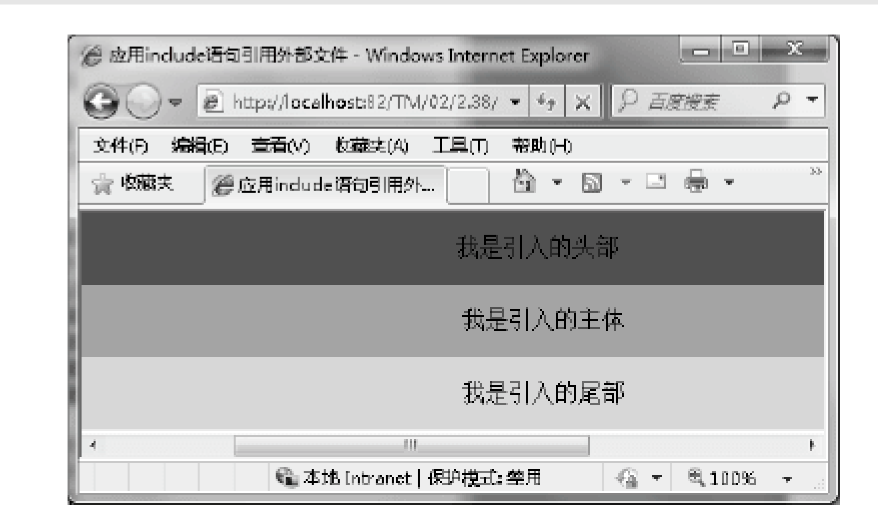


图 2.22　应用 include 语句引用文件

2.8.2 应用 require 语句引用文件

require 语句的使用方法与 include 语句类似，都是实现对外部文件的引用。在 PHP 文件被执行前，PHP 解析器会用被引用的文件的全部内容替换 require 语句，然后与 require 语句之外的其他语

句组成新的 PHP 文件，最后再按新 PHP 文件执行程序代码。

注意 因为 require 语句相当于将另一个源文件的内容完全复制到本文件中，所以一般将其放在源文件的起始位置，用于引用需要使用的公共函数文件和公共类文件等。

PHP 可以使用任何扩展名来命名引用文件，如.inc 文件、.html 文件或其他非标准的扩展名文件等，但 PHP 通常用来解析扩展名被定义为.php 的文件。建议读者使用标准的文件扩展名。

语法：

```
void require(string filename);
```

参数 filename 是指定的完整路径文件名。

例 2.39 应用 require 语句引用并运行指定的外部文件 top.php。**（实例位置：光盘\TM\Instance\02\2.39）**

代码如下：

```
<?php
require("top.php");                                    //嵌入外部文件 top.php 页
?>
```

top.php 文件的代码如下：

```
<table width="750" height="131" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td  bgcolor="#FFCCCC"  align="center">明日科技有限公司是一家知名的软件公司，公司的网址是
www.mingribook.com</td>
  </tr>
</table>
```

运行 index.php 页，输出显示了 require 语句引用的 Web 页。运行结果如图 2.23 所示。

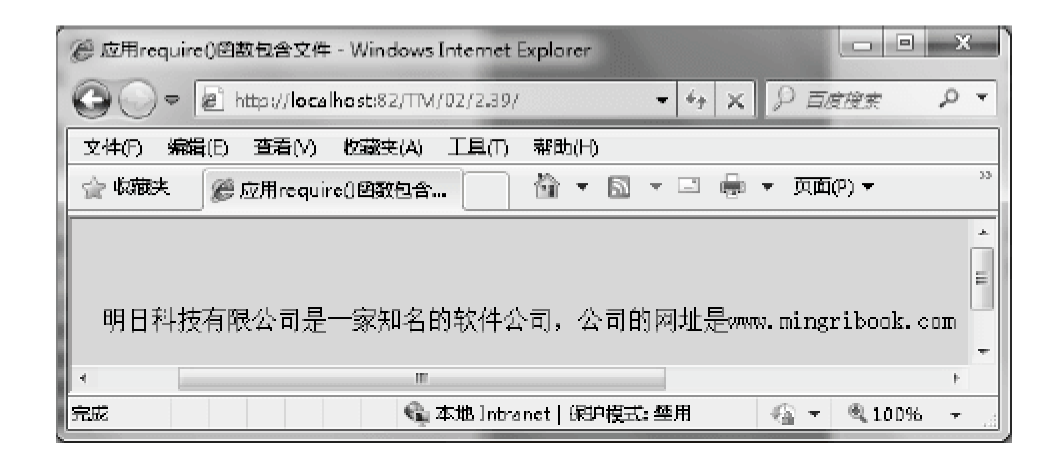


图 2.23 应用 require 语句引用外部文件

2.8.3 应用 include_once 语句引用文件

在使用 include_once 语句时，应该明确它与 include 语句的区别，应用 include_once 语句会在导入文件前先检测该文件是否在该页面的其他部分被引用过，如果有，则不会重复引用该文件，程序只能引用一次。例如，要导入的文件中存在一些自定义函数，那么如果在同一个程序中重复导入这个文件，在第二次导入时便会发生错误，因为 PHP 不允许相同名称的函数被重复声明。

语法：

```
void include_once (string filename);
```

参数 filename 是指定的完整路径文件名。

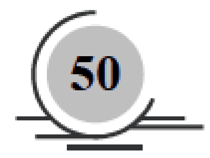
例 2.40 应用 include_once 语句引用并运行指定的外部文件 top.php。**（实例位置：光盘\TM\Instance\02\2.40）**

代码如下：

```
<?php
include_once("top.php");                               //嵌入外部文件 top.php 页
?>
```

top.php 文件的代码如下：

```
<table width="779" height="80" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td   bgcolor="#33CCFF">使用 include_once 语句引用的文件</td>
  </tr>
</table>
```



运行结果如图 2.24 所示。

2.8.4　应用 require_once 语句引用文件

require_once 语句是 require 语句的延伸，它的功能与 require 语句基本类似，不同的是，在应用 require_once 语句时会先检查要引用的文件是不是已经在该程序中的其他地方被引用过，如果有，则不会再次重复调用该文件。例如，同时应用 require_once 语句在同一页面中引用了两个相同的文件，那么在输出时只有第一个文件被执行，第二次引用的文件不会被执行。

语法：

```
void require_once (string filename);
```

参数 filename 是指定的完整路径文件名。

例 2.41　下面讲解应用 require_once 语句调用外部文件的实现方法。**（实例位置：光盘\TM\Instance\02\2.41）**

（1）首先创建一个简单 php 页，命名为 index.php，然后应用 require_once 语句嵌入 3 个外部文件。代码如下：

```
<table width="1004" border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td><?php require_once("top.php");?></td>                    //嵌入头文件
  </tr>
  <tr>
    <td><?php require_once("main.php");?></td>                   //嵌入主文件
  </tr>
  <tr>
    <td><?php require_once("bottom.php");?></td>                 //嵌入尾文件
  </tr>
</table>
```

（2）然后在外部文件 main.php 页中动态显示相关的数据信息。运行结果如图 2.25 所示。

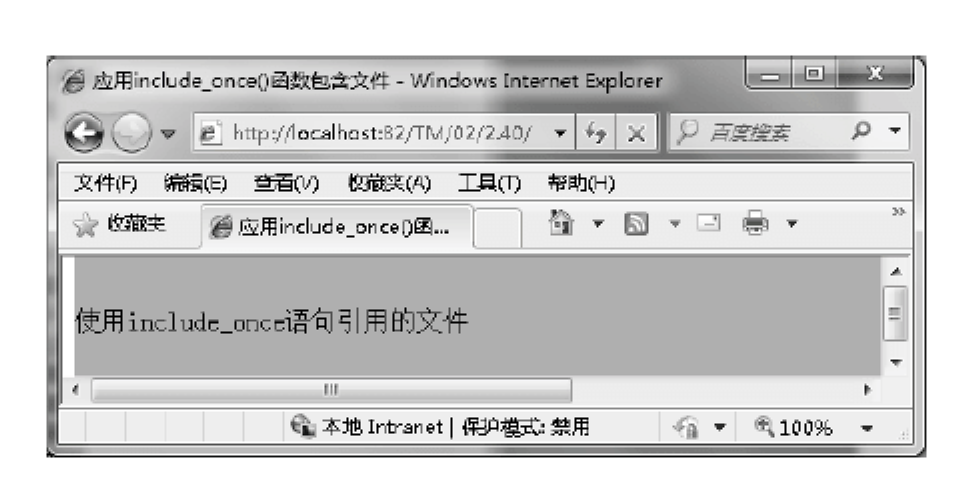


图 2.24　应用 include_once 语句引用文件

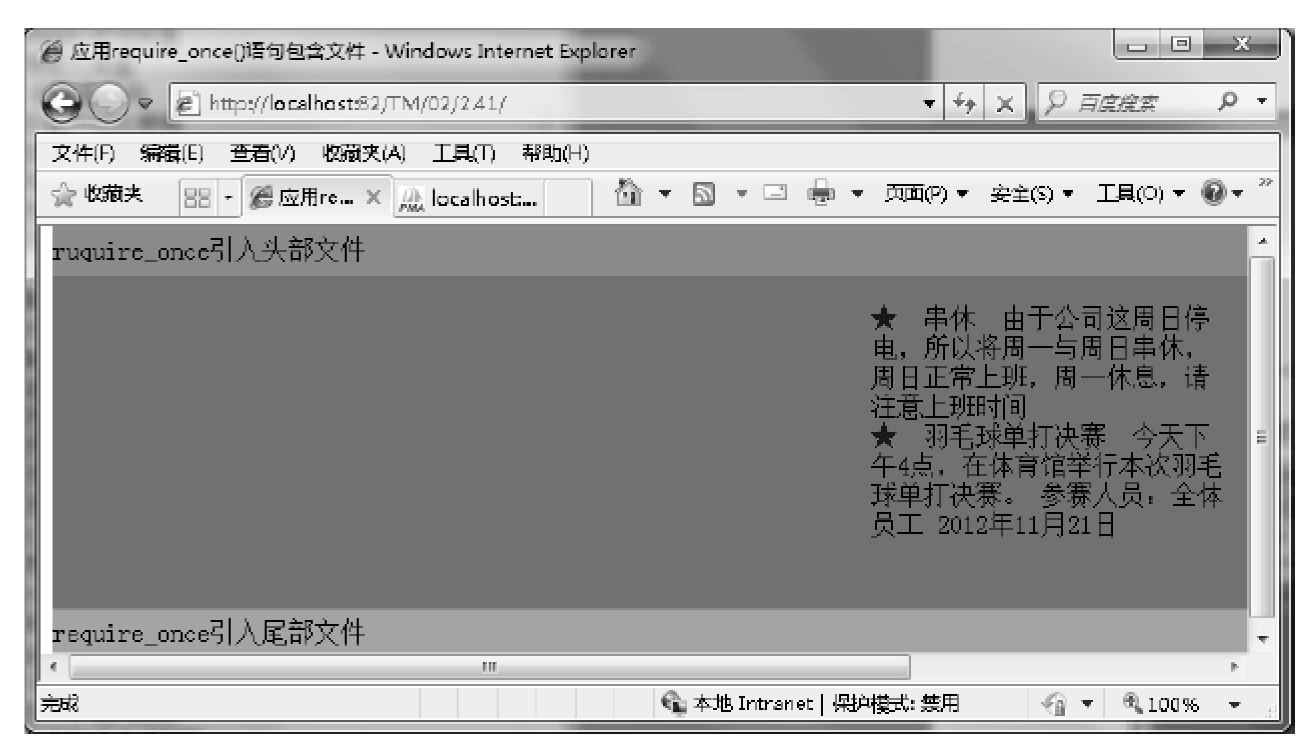


图 2.25　应用 require_once 语句引用文件

2.8.5　include 语句和 require 语句的区别

应用 require 语句来调用文件，其应用方法和 include 语句是类似的，但存在如下区别。

☑ 在使用 require 语句调用文件时，如果调用的文件没找到，require 语句会输出错误信息，并且立即终止脚本的处理。而 include 语句在没有找到文件时则会输出警告，不会终止脚本的处理。

☑ 使用 require 语句调用文件时，只要程序一执行，就会立刻调用外部文件；而通过 include 语句调用

外部文件时，只有程序执行到该语句时，才会调用外部文件。

2.8.6　include_once 语句和 require_once 语句的区别

include_once 语句和 require_once 语句的用途是确保一个被包含文件只能被包含一次。使用这两个语句可以防止意外地多次包含相同的函数库，从而减少函数重复定义并产生错误。

但两者之间是有区别的：include_once 语句在脚本执行期间调用外部文件发生错误时，产生一个警告，而 require_once 语句则导致一个致命错误。

2.9　实　　战

视频讲解：光盘\TM\Video\第 2 章\实战.exe

2.9.1　判断闰年的方法

例 2.42　下面应用运算符和表达式来判断用户指定的年份是否是闰年。闰年的判断方式是，能够被 4 整除不能够被 100 整除，或者能够被 100 整除并且能被 400 整除的年份为闰年。（**实例位置：光盘\TM\Instance\02\2.42**）

本实例的完整代码如下：

```
<table width="392" height="101" border="1" align="center" cellpadding="1" cellspacing="1" bordercolor="#009900"
bgcolor="#CCFF00">
  <tr>
  <td height="25" colspan="2" align="center" class="STYLE5">判断指定的年份是否为闰年</td>
  </tr>
    <tr>
      <td width="160" height="30" bgcolor="#FFFF99"><span class="STYLE3">请输入一个四位数的年份：
</span></td>
      <td width="219" bgcolor="#FFFF99">
<form name="form1" method="post" action="">
       <table width="196" height="29" border="0" cellpadding="0" cellspacing="0">
         <tr>
           <td width="136"><input name="txt_year" type="text" value="" size="20">          </td>
           <td width="60"> 
             <input type="submit" name="Button" value="计算"></td>
         </tr>
      </table>
</form>
              </td>
    </tr>
    <tr>
      <td height="37" colspan="2" align="center">
<?php
if($_POST[Button]!=""){
       $year=$_POST[txt_year];
       $result =($year%4==0 && $year%100!=0)||($year%400==0)?$year."是闰年":$year."不是闰年";
       echo $result;
}
```

```
?>
            </td>
   </tr>
</table>
```

运行本实例，在文本框中输入指定的年份，单击“计算”按钮，即可求出该年份是否为闰年。运行结果如图 2.26 所示。

2.9.2　通过自定义函数防止新闻主题信息出现中文乱码

例 2.43　创建一个自定义函数，用于实现屏蔽中文乱码的输出，将该函数封装在一个 function.php 文件中，然后应用 include_once 语句引用这个文件，再通过 echo 语句输出新闻主题信息，截取前 18 个字符，并应用自定义函数屏蔽中文乱码。**（实例位置：光盘\TM\Instance\02\2.43）**

（1）为了保持整个页面的合理布局，经常需要对一些较长的字符进行部分输出，但由于汉字占有两个字符，如果截取位置选取不当就可能导致截取的字符串末尾出现乱码，本例将解决上述问题。

运行本实例，结果如图 2.27 所示，无论在何处对字符串进行截取都不会出现乱码。




图 2.26　判断指定的年份是否为闰年

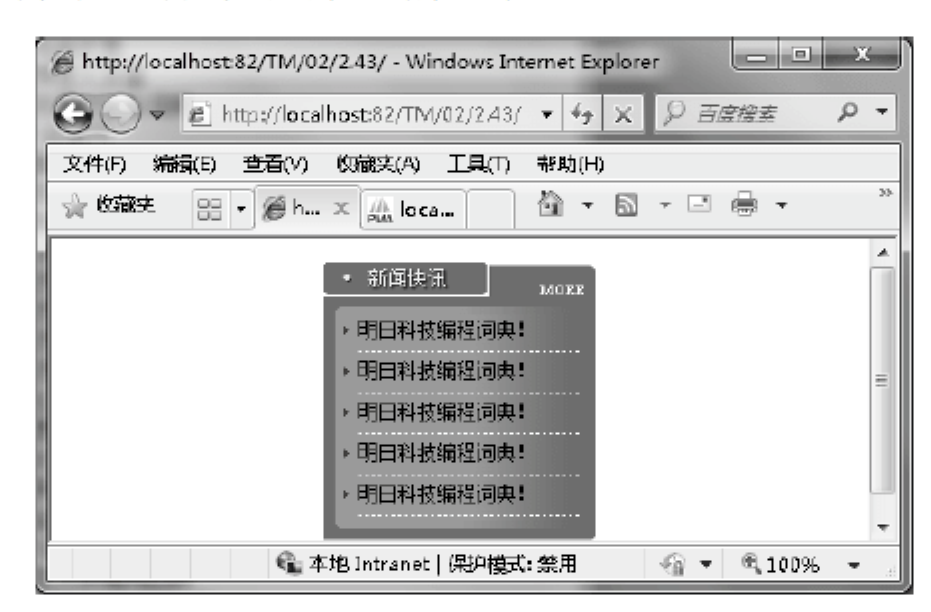


图 2.27　新闻主题信息显示

（2）PHP 对字符串进行截取可以采用 PHP 的预定义函数 substr()实现。该函数的使用方法如下：

```
string substr ( string string, int start [, int length])
```


说明　该函数对 string 进行截取操作，截取从字符串的 start 位置开始，到 length 长度结束。

应用 substr()函数对普通字符进行截取操作会非常方便，但对全角字符进行截取时可能会造成乱码出现。为了解决该问题可以通过自定义函数 chinesesubstr()解决上述问题。该函数代码如下：

```
<?php
 function chinesesubstr($str,$start,$len) {
     $strlen=$start+$len;
     for($i=0;$i<$strlen;$i++) {
          if(ord(substr($str,$i,1))>0xa0) {
               $tmpstr.=substr($str,$i,2);
               $i++;
          } else
               $tmpstr.=substr($str,$i,1);
     }
     return $tmpstr;
}
?>
```

该函数中的参数$str 表示要截取的字符串，$start 表示从何位置开始截取，$len 表示截取字符串的长度，

考虑到编程人员的习惯，自定义函数 chinesesubstr()与 PHP 预定义函数 substr()的参数设计一致，这样程序员就可以像使用函数 substr()一样使用 chinesesubstr()了。这里建议将 chinesesubstr()函数单独放入某文件，使用时用 include()函数包含该文件即可。

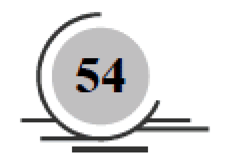
（3）引用自定义函数 chinesesubstr()所在的 function.php 文件，并通过循环显示公告主题的前 16 个字符。

```
<table width="160" border="0" align="center" cellpadding="0" cellspacing="0">
  <tr>
     <td width="165" height="32"><img src="images/tell_top.gif" width="165" height="32" border="0" ></td>
        </tr>
        <tr>
          <td height="52" background="images/tell_center.gif">
            <?php
                 include_once("function.php");
                 $news="明日科技编程词典！ ";
                 $i=1;
                 do{
              ?>
            <table width="148" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
              <tr>
                <td width="17" height="20"><img src="images/mark_0.gif" width="10" height="10"></td>
                <td width="333">
             <?php
                echo chinesesubstr($news,0,18);
                if(strlen($news)>18){
                         echo " ...";
                }
                ?>
                  </td>
              </tr>
              <tr >
                <td height="5"></td>
                <td height="5"   background="images/back_point_write.gif"></td>
              </tr>
            </table>
              <?php
               $i++;
               }while($i<=5);
              ?>
              </td>
        </tr>
        <tr>
          <td width="165" height="12"><img src="images/tell_bottom.gif" width="165" height="12"></td>
        </tr>
      </table>
```

说明 在 PHP 动态网页开发的过程中，从数据库读取数据最为常用，由于数据库的内容在后面的章节中进行讲解，因此，本例使用了字符串变量替代数据库数据，在数据库章节中会讲解如何读取数据库中的内容。

2.9.3 应用 include 语句构建在线音乐网站主页

例 2.44 本实例应用 include 语句引用外部文件来构建网站主页。要使网站主页的代码简洁化，而且维



护起来很方便，只需要在主页应用表格制作一个简练、大气、个性鲜明的网页布局，然后将每个布局区域内的内容封装成一个独立的文件，最后应用 include 语句引用这些独立的文件即可轻松构建主页。下面讲解应用 include 语句引用外部文件的方法。（**实例位置：光盘\TM\Instance\02\2.44**）

对于一个网站而言，主页作为直接的信息展台，所承载的信息量将是巨大的。从网站程序编写来说，编写首页的代码需要几百行，甚至上千行，这对于后期维护来说是很麻烦的，尤其是当其他管理员进行维护时，由于对代码不熟悉，经常需要在几千行代码中来回查找。那么，如何来解决这种问题呢？应用引用文件是解决该问题最有效的方法。将拥有指定功能的代码封装成一个单独的文件，当用到该文件时，只需要使用 include 等引用语句对该文件进行引用即可。这样做的目的在于使得对页面的管理和维护都变得更简洁、方便，并且减少了代码的重用性。

本实例的实现方法如下：

（1）首先创建一个简单 PHP 动态页，命名为 index.php。

（2）设计一个 3 行 1 列的表格，并将第二行拆分成两个单元格。然后应用 include 语句引用 4 个外部文件，分别是 top.html 页、left.html 页、main.html 页和 bottom.html 页。构建网站主页布局的效果如图 2.28 所示。

在 index.php 文件中应用 include 语句引用 4 个外部文件的代码如下：

```
<table width="768" border="0" align="center" cell padding="0" cellspacing="0">
  <tr>
    <td colspan="2"><?php include("top.html");?></td>
  </tr>
  <tr>
    <td width="180"><?php include("left.html"); ?></td>
    <td width="588"><?php include("main.html"); ?></td>
  </tr>
  <tr>
    <td colspan="2"><?php incluce("bottom.html");?></td>
  </tr>
</table>
```

（3）在引用的其他子页中应用 HTML 标记设计子页内容。运行结果如图 2.29 所示。

top.html
left.html
main.html
bottom.html

图 2.28　构建网站主页布局

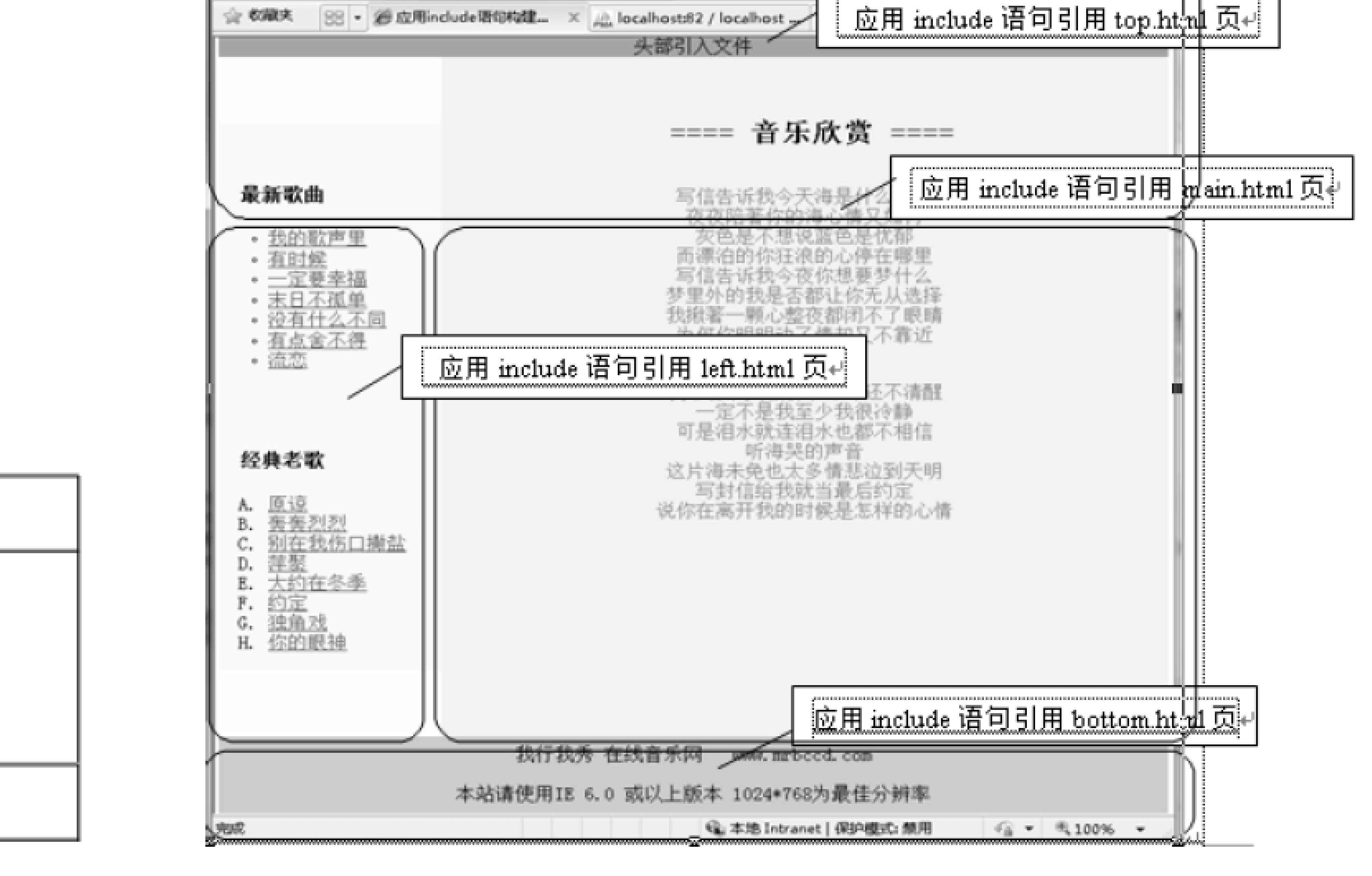
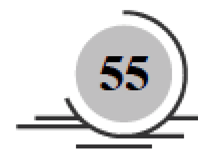


图 2.29　应用 include 语句构建网站主页



说明 在 PHP 动态网页开发的过程中，在功能上使用最频繁的就是对数据库连接文件的引用。将数据库连接文件封装在一个独立的文件中，然后应用 include 语句或者 require 语句等对封装的文件进行引用，由于数据库内容将在后面的章节中进行介绍，因此，引用数据库文件的方法将在后面进行详细介绍。

2.9.4 随机组合生日祝福语

例 2.45 创建 index.php 文件，首先定义两个数组，将字符串信息保存在数组中。然后，利用 rand 函数随机取得数组中的两条信息并保存在变量中。（**实例位置：光盘\TM\Instance\02\2.45**）

本实例的完整代码如下：

```
<?php
	$arr = array("生日快乐","今天是你的出生日","同学们为你许愿");
	$arr1 = array("祝你万事如意","祝你生日快乐","祝你每天都有一份好心情");
	$rand = rand(0,2);
	echo $arr[$rand].$arr1[$rand];
?>
```

运行结果如图 2.30 所示。

2.9.5 加法计算器

例 2.46 下面制作一个简单的加数器，当用户单击“等于”按钮时，通过算术运算符“+”计算两个文本框中数据的和，并显示在文本框中。（**实例位置：光盘\TM\Instance\02\2.46**）

（1）首先，创建 index.php 文件，设计表单页面，效果如图 2.31 所示。

（2）获取表单中提交的数据，完成加法运算。代码如下：

```
<?php
if(isset($_POST['Submit']) and $_POST['number1']!=null and $_POST['number2']!=null and $_POST['Submit']=="等于"){
	$number3=$_POST['number1']+$_POST['number2'];			//计算变量值
	}else{
	$number3=null;
	}
?>
  <label>
  <input name="number3" type="text" id="number3" size="10" value="<?php echo $number3;?>"/> <!--输出变量值-->
  </label>
```

运行结果如图 2.31 所示。

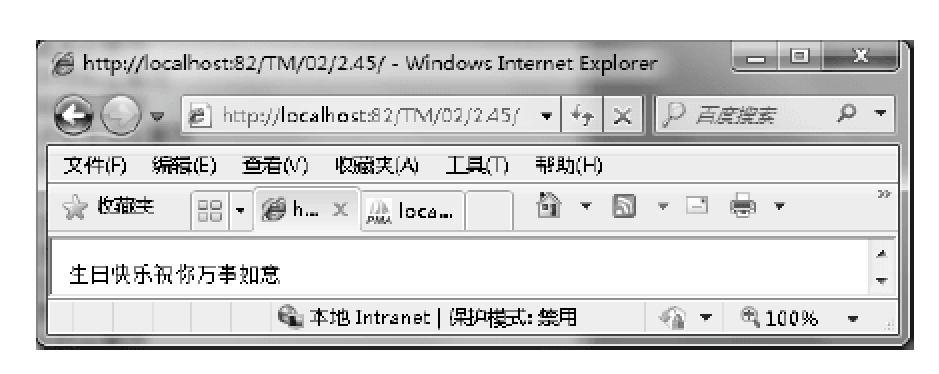


图 2.30 随机获取内容

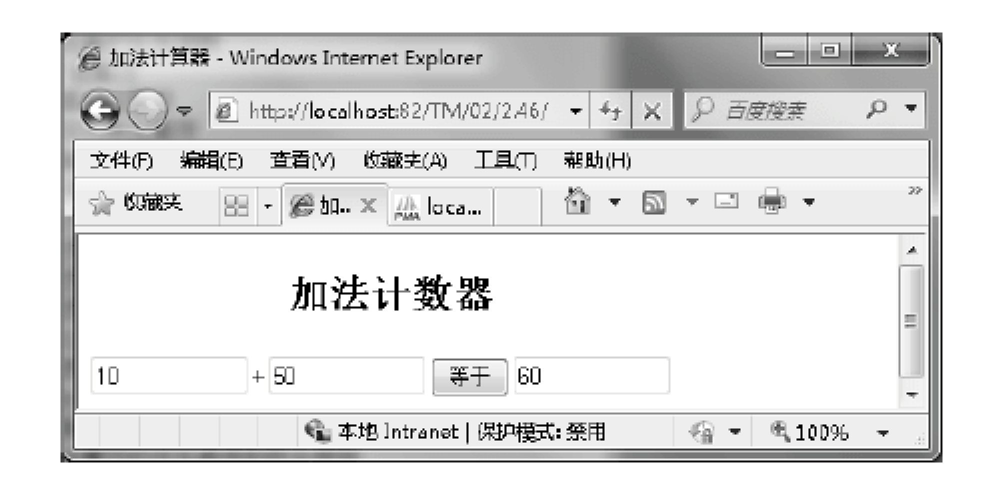
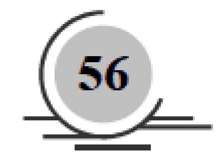


图 2.31 加法计算器



2.10　小　　结

本章主要介绍了 PHP 语言的基础知识，重点讲解了 PHP 的语法基础、数据类型、常量、变量、运算符和表达式，以及 PHP 函数、输出语句等。基础知识是一门语言的核心，希望初学者能静下心来，牢牢掌握本章的知识。只有基础扎实，才能真正迈过精通 PHP 的门槛。

2.11　学习成果检验

1．应用字符串运算符输出如图 2.32 所示的结果。**（答案位置：光盘\TM\Instance\02\2.47）**

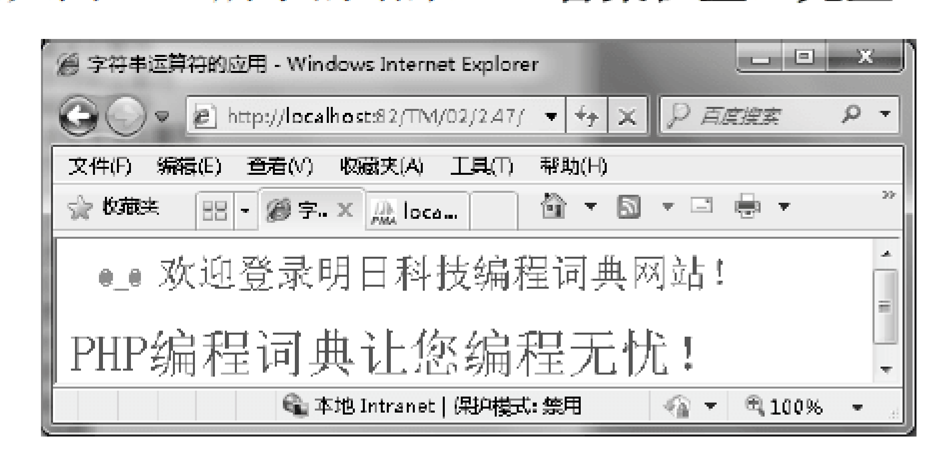


图 2.32　字符串运算符的应用

2．在开发动态网站中的用户注册模块时，其中用户注册的密码一般都要填写一个确认密码，来加强输入信息的准确性。下面在用户注册模块中，应用比较运算符验证两次输入的密码是否一致。如果相等则输出注册成功信息，否则将输出错误提示信息。**（答案位置：光盘\TM\Instance\02\2.48）**

3．PHP 的输出语句有 echo、print、printf、print_r。尝试使用这 4 个语句分别输出数据，看它们之间有什么不同。**（答案位置：光盘\TM\Instance\02\2.49）**

第 3 章

PHP 流程控制语句

（视频讲解：54 分钟）

条件控制语句和循环控制语句是两种基本的语法结构，它们都是用来控制程序执行流程的，也是构成程序的主要语法基础。通常，语句是按顺序执行的，即执行完当前的语句后，执行下一条语句。但是这种简单的顺序执行方式能够解决的问题很有限，现实生活中的许多逻辑使用顺序执行的方式是不能够被描述的。因此，开发人员可以按照实际问题的逻辑思路选择性地使用流程控制语句来编写程序代码。

通过阅读本章内容，你可以：

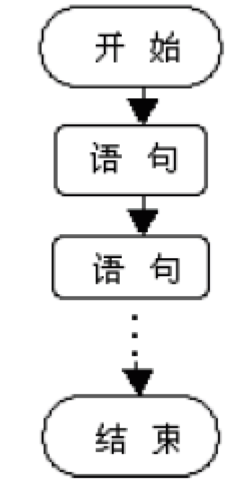
- 了解 IF 语句的使用
- 了解 switch…case 多重判断语句
- 掌握 while 循环语句
- 掌握 do…while 循环语句
- 掌握 for 循环语句
- 掌握 foreach 循环语句
- 了解 break/continue 关键字

3.1　控 制 结 构

在编程的过程中，所有的操作都是在按照某种结构有条不紊地进行，纠其根源，程序的控制结构大致可以分为 3 种：顺序结构、选择结构和循环结构。在对这 3 种结构的使用中，几乎很少有哪个程序是单独使用某一种结构来完成某个操作的，基本上都是其中的两种或者 3 种结构结合使用。为了帮助读者更好地应用这 3 种程序控制结构，下面对其进行详细介绍。

1．顺序结构

顺序结构是最简单、最基本的结构方式，各流程框依次按顺序执行。其传统流程图的表示方式与 N-S 结构化流程图的表示方式分别如图 3.1 和图 3.2 所示。其执行顺序为：开始→语句 1→语句 2→…→结束。



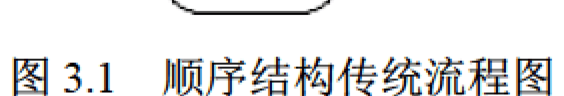

图 3.1　顺序结构传统流程图



图 3.2　N-S 结构化流程图

2．选择（分支）结构

选择结构就是对给定条件进行判断，条件为真时执行一个分支，条件为假时执行另一个分支。其传统流程图表示方式与 N-S 结构化流程图表示方式分别如图 3.3 和图 3.4 所示。

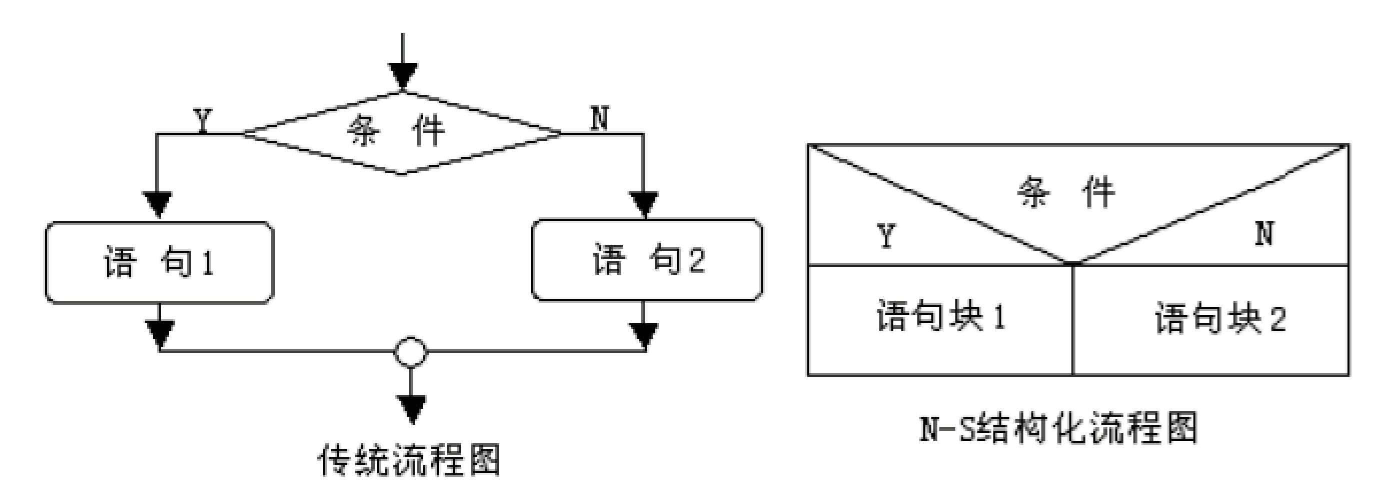


图 3.3　条件成立与否都执行语句或语句块

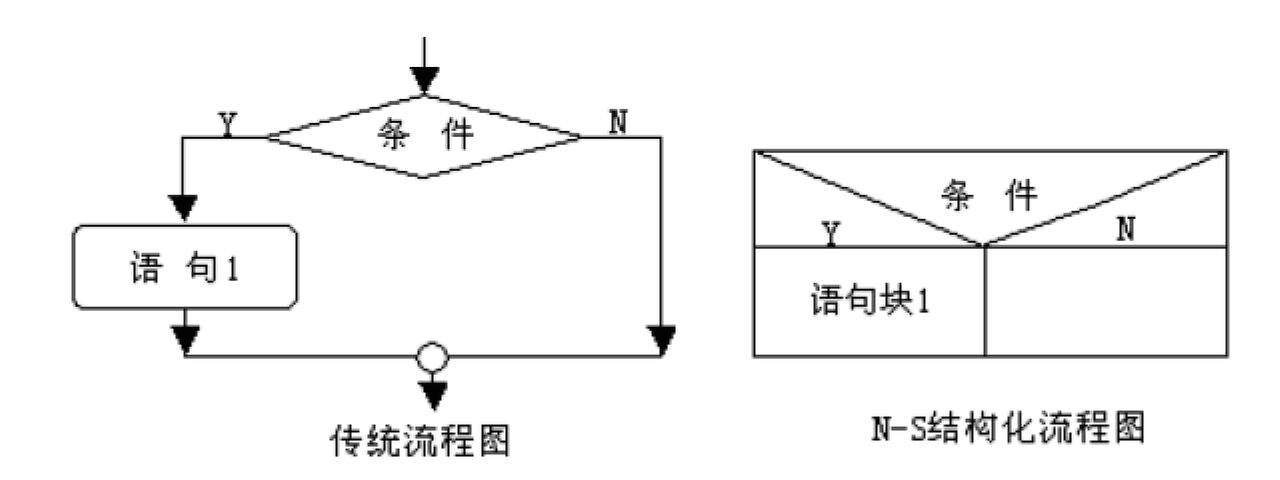


图 3.4　条件为否则不执行语句或语句块

3．循环结构

循环结构可以根据需要多次重复执行一行或者多行代码。循环结构分为两种：前测试型循环和后测试

型循环。

前测试型循环，先判断后执行。当条件为真时反复执行语句或语句块，条件为假时，跳出循环，继续执行循环后面的语句，流程图如图 3.5 所示。

后测试型循环，先执行后判断。先执行语句或语句块，再进行条件判断，直到条件为假时，跳出循环，继续执行循环后面的语句，否则一直执行语句或语句块，流程图如图 3.6 所示。

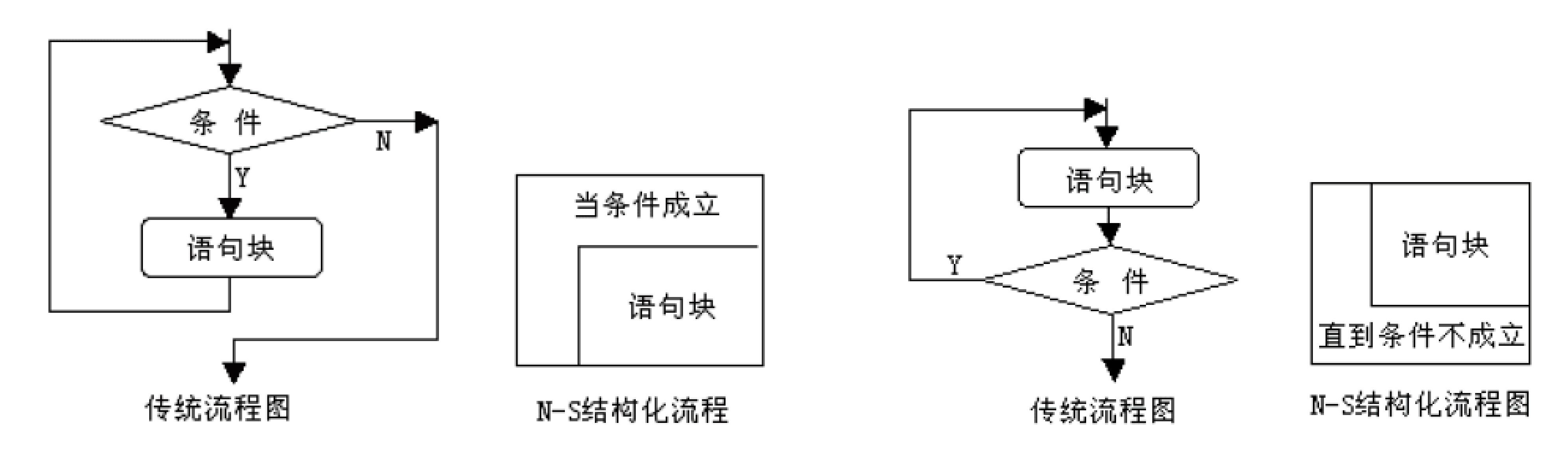


图 3.5　前测试型循环流程图　　图 3.6　后测试型循环流程图

在 PHP 中，大多数情况下程序不会是简单的顺序结构，而是以这 3 种结构的组合形式出现。其中的顺序结构很容易理解，就是直接输出程序运行结果，而选择和循环结构则需要以下 3 种控制语句来实现。

☑ 条件控制语句：if、else、elseif 和 switch 语句。

☑ 循环控制语句：while、do…while、for 和 foreach 语句。

☑ 跳转控制语句：break 和 continue 语句。

在下面的章节中将对这 3 种控制语句进行详细讲解。

3.2　条件控制语句

所谓条件控制语句就是对语句中不同条件的值进行判断，进而根据不同的条件执行不同的语句。在条件控制语句中主要有两个语句：if 条件控制语句和 switch 多分支语句。

3.2.1　if 条件控制语句

视频讲解：光盘\TM\Video\第 3 章\if 条件控制语句.exe

1．if 语句

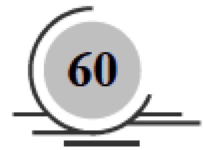
if 语句是最简单的条件判定语句，它对某段程序的执行附加一个条件，如果条件成立，就执行这段程序；否则就跳过这段程序去执行下面的程序。

if 语句的格式为：

```
if (expr)
    statement ;
```

如果表达式 expr 的值为 true，那么就顺序执行 statement 语句；否则，就跳过该条语句再往下执行。如果需要执行的语句不只一条，那么可以使用“{ }”（在“{ }”中的语句，被称之为语句组）。其格式为：

```
if(expr){
      statement1;
      statement2;
      …
}
```



if 条件语句的流程图如图 3.7 所示。

例 3.1　在本例中，实现输出“如果元旦放假，我们一起去爬山”。首先为变量赋一个逻辑值，然后判断这个值是否为真，如果为真，则输出结果。（**实例位置：光盘\TM\Instance\03\3.1**）

代码如下：

```
<?php
	$day_51=true;						//为变量赋予一个逻辑值
	if ($day_51==true){					//判断变量的逻辑值是否为真
		echo "如果元旦放假，我们一起去爬山";		//输出字符串
	}
?>
```

运行结果如图 3.8 所示。

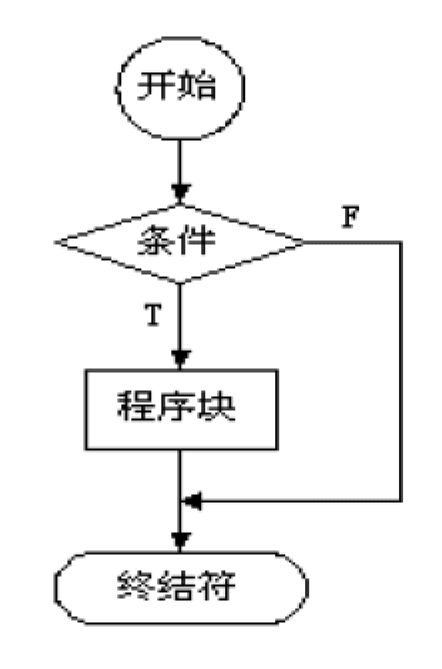


图 3.7　if 语句流程图

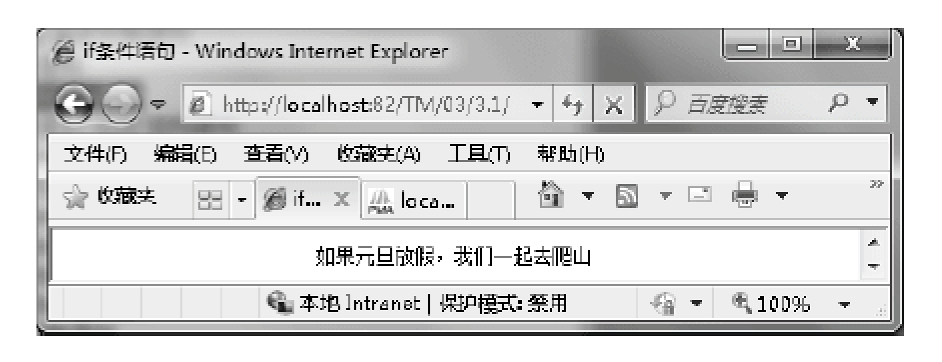


图 3.8　if 语句的应用

2．if…else 语句

大多数时候，总是需要在满足某个条件时执行一条语句，而在不满足该条件时执行其他语句。为了在 if 语句中描述这种情况，if 语句中提供了 else 子句，else 子句表示的自然语句是“否则”的意思。其语法格式为：

```
if(expr){
	statement1;
}else{
	statement2;
}
```

当表达式 expr 的值为 true 时，执行 statement1 语句；如果表达式 expr 的值为 false，则执行 statement2 语句。if…else 语句的流程图如图 3.9 所示。

例 3.2　在许多问题的逻辑关系中，经常会出现依据一个条件是否成立会导致两种不同的结果。例如，如果元旦放假，我们一起去爬山，否则我们在家看电视。首先为变量赋一个逻辑值，然后判断这个值是否为 true，如果为 true，则输出“如果元旦放假，我们一起去爬山”；否则输出“我们在家看电视”，根据不同的结果显示不同的字符串。（**实例位置：光盘\TM\Instance\03\3.2**）

实例代码如下：

```
<?php
	$day_51=false;						//为变量赋予一个逻辑值
	if ($day_51==true){					//判断变量的逻辑值是否为真
		echo "如果元旦放假，我们一起去爬山";		//输出字符串
	}
	else{
		echo "我们在家看电视";				//输出字符串
	}
?>
```

结果为：我们在家看电视。

3．嵌套的 else if 结构语句

使用 if 语句和 else 子句能够描述一些复杂的逻辑问题，但是有时并不能够完整地表达人们的语义。考虑这样一种情况，if…else 语句只能选择两种结果：要么执行真，要么执行假。但如果现在有两种以上的选择该怎么办呢？例如，小红的考试成绩，如果是 90 分以上，则为“优秀”；如果是 60～90 分之间，则为“良好”；如果低于 60 分，则为“不及格”。这时，可以使用 else if（也可以写做 else if）语句来执行。该语法格式为：

```
if(expr1){
      statement 1;
}else if(expr2){
      statement 2;
}…
else{
      statement n;
}
```

else if 语句的流程图如图 3.10 所示。

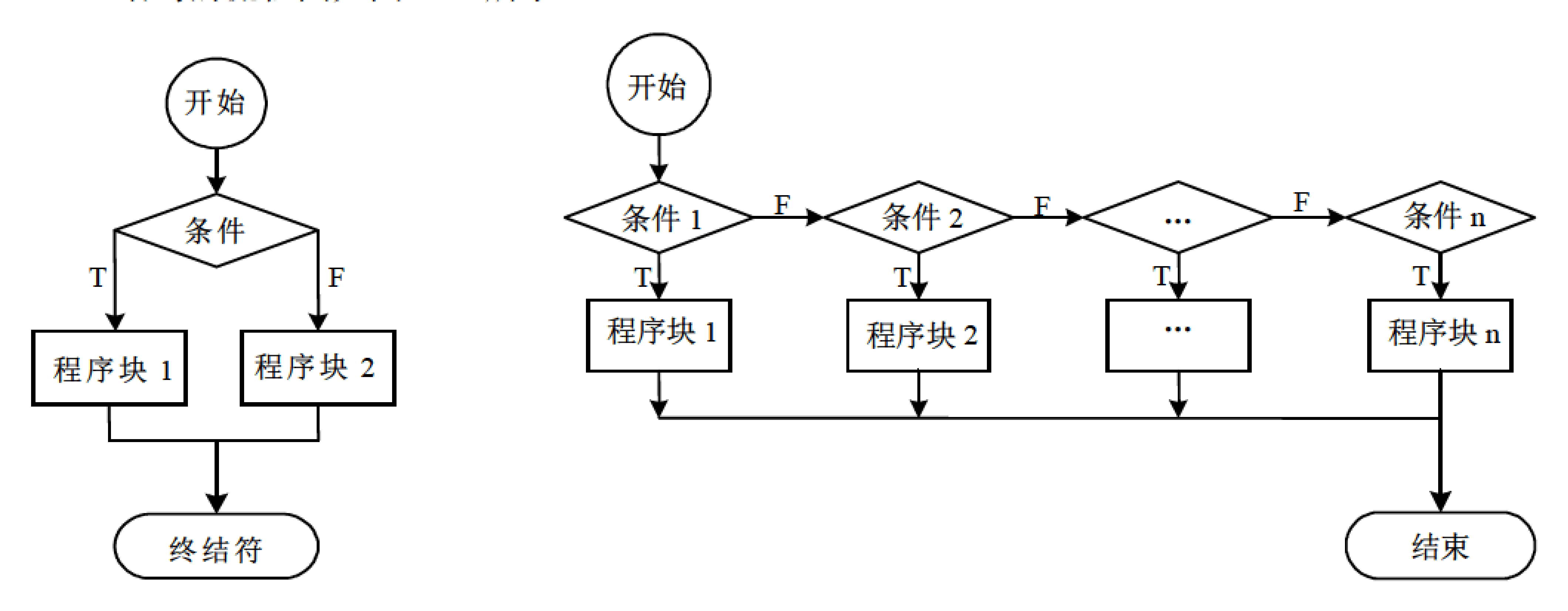


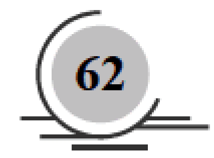
图 3.9　if…else 语句流程图　　　　图 3.10　else if 语句的流程图

例 3.3　通过 else if 语句，判断小红的考试情况。如果成绩大于等于 90，为优秀；如果成绩大于等于 80，则为良好；如果成绩大于等于 60，则为及格，否则为不及格。**（实例位置：光盘\TM\Instance\03\3.3）**

实例代码如下：

```
<?php
      $score=95;                                  //设置小红期末考试的默认值
      if ($score >=90){                           //判断小红的期末考试成绩是否在 90 分以上
            echo "小红期末考试成绩优秀";            //如果是，说明小红期末考试成绩优秀
      }elseif($score<90 && $score>=80){           //否则判断小红期末考试成绩是否在 80～90 之间
            echo "小红期末考试成绩良好";            //如果是，说明小红期末考试成绩良好
      }elseif($score<80 && $score>=60){           //否则判断小红期末考试成绩是否在 60～80 之间
            echo "小红期末考试成绩及格";            //如果是，说明小红期末考试成绩及格
      }else{                                      //如果两个判断都是 false，则输出默认值
            echo "小红期末考试成绩不及格";          //说明小红期末考试成绩不及格
      }
?>
```

结果为：小红期末考试成绩优秀。



注意 if 语句和 else if 语句的执行条件是表达式的值为真，而 else 执行条件是表达式的值为假。这里表达式的值不等于变量的值。

3.2.2　switch…case 分支控制语句

视频讲解：光盘\TM\Video\第 3 章\switch…case 分支控制语句.exe

在程序设计中，所有依据条件作出判定的问题，都可以用前面所介绍的不同类型的 if 语句来解决。不过，在用 if…else 语句处理多个条件的判定问题时，组成条件的表达式在每一个 else if 语句中都要计算一次，显得繁琐臃肿。为了避免 if 语句的冗长，提高程序的可读性，可以使用 switch 分支控制语句。PHP 提供 switch…case 多分支控制语句，对于某些多项选择场合，会使程序代码更加简洁、易读。

switch 语句的语法格式如下：

```
switch(variable){
	case value1:
		statement1;
		break;
	case value2:
	…
	default:
		default statement n;
}
```

switch 语句根据 variable 的值，依次与 case 中的 value 值相比较，如果不相等，继续查找下一个 case；如果相等，就执行对应的语句，直到 switch 语句结束或遇到 break 为止。一般 switch 语句的结尾都有一个默认值 default，如果在前面的 case 中没有找到相符合的条件，则输出默认语句，这和 else 语句类似。

switch 语句的流程图如图 3.11 所示。

例 3.4　将使用 switch 语句来输出当天为星期几，并根据不同的日期输出不同的提醒警句。(**实例位置：光盘\TM\Instance\03\3.4**)

实例代码如下：

```
<?php
setlocale(LC_TIME,"chs");                          //设置本地环境
$weekday = strftime("%A");                         //声明变量$weekday 的值
switch ($weekday){                                 //switch 语句，判断$weekday 的值
	case "星期一":                                 //如果变量的值为“星期一”
		echo "今天是$weekday ，新的一天开始了！";
		break;
	case "星期二":                                 //如果变量的值为“星期二”
		echo "今天是$weekday ，认真的工作态度真的很重要!";
		break;
	case "星期三":                                 //如果变量的值为“星期三”
		echo "今天是$weekday ，充实生活，努力工作！";
		break;
	case "星期四":                                 //如果变量的值为“星期四”
		echo "今天是$weekday ，勤奋才能创造绩效，加油！）";
		break;
	case "星期五":                                 //如果变量的值为“星期五”
		echo "今天是$weekday ，积极完成工作任务！";
		break;
	case "星期六":                                 //如果变量的值为“星期六”
```

```
            echo "今天是$weekday ，可以放松心情了！";
            break;
        default:                                                    //默认值
            echo "今天是$weekday，去 happy 一下！";
            break;
}
?>
```

运行结果如图 3.12 所示。

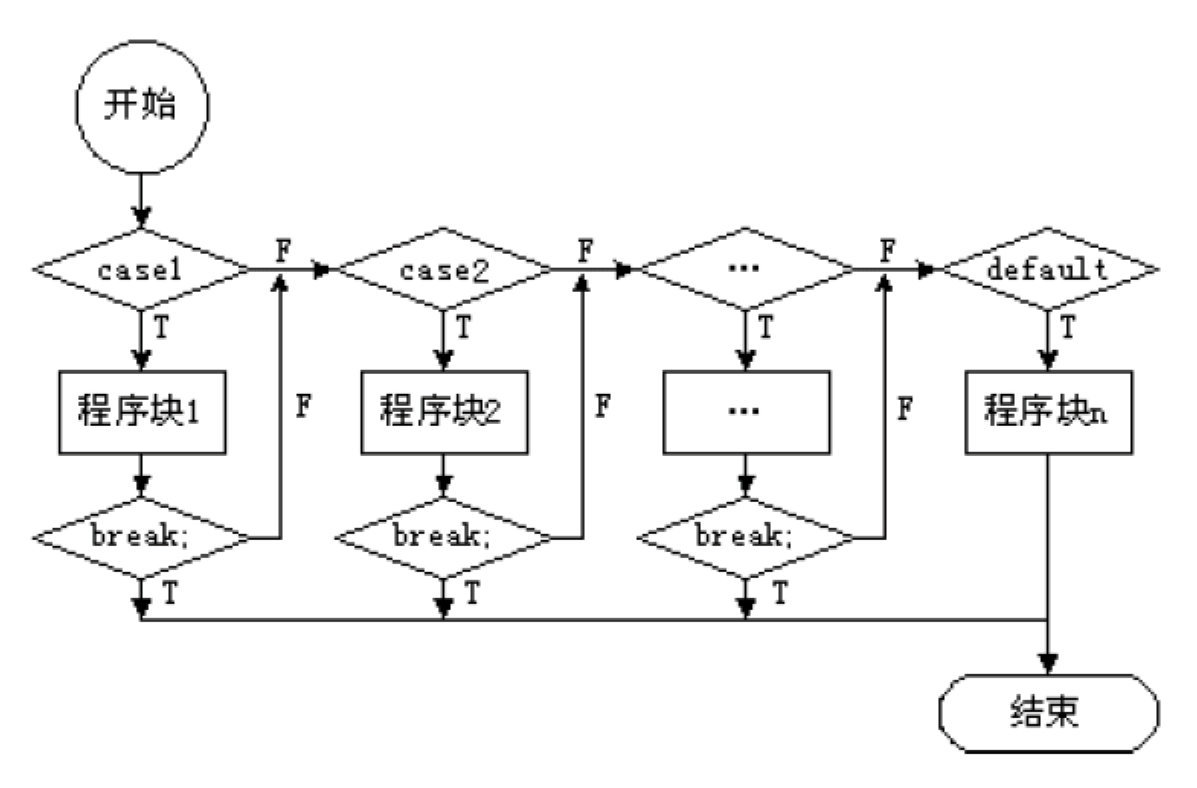


图 3.11 switch 语句流程图

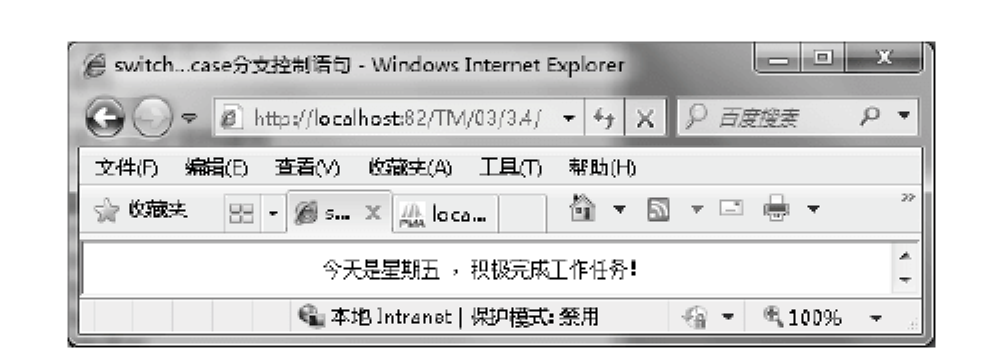


图 3.12 switch…case 分支控制语句的应用

注意 switch 语句在执行时，即使遇到符合要求的 case 语句段，也会继续往下执行，直到 switch 结束。为了避免这种浪费时间和资源的情况发生，一定要在每个 case 语句段后添加 break 跳转语句跳出当前循环。break 跳转语句将在本章 3.4.1 节中进行详细介绍。

3.3 循环控制语句

循环语句是在满足条件的情况下反复执行某一个操作。在 PHP 中，提供 4 个循环控制语句，分别是 while 循环语句、do…while 循环语句、for 循环语句和 foreach 循环语句。

3.3.1 while 循环语句

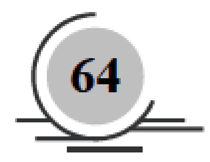
视频讲解：光盘\TM\Video\第 3 章\while 循环语句.exe

while 循环语句，其作用是反复执行某一项操作，是循环控制语句中最简单的一个，也是最常用的一个。while 循环语句对表达式的值进行判断，当表达式为非 0 值时，执行 while 语句中的内嵌语句；当表达式的值为 0 值时，则不执行 while 语句中的内嵌语句。该语句的特点是：先判断表达式，后执行语句。while 循环控制语句的操作流程如图 3.13 所示。

其语法如下：

```
while (expr){                               /*
        statement;                          先判断条件，当条件满足时执行语句块，否则不向下执行
}                                           */
```

只要 while 表达式 expr 的值为 true，就重复执行嵌套中的 statement 语句，如果 while 表达式的值一开始



就是 false，则循环语句一次也不执行。

例 3.5　将 10 以内的偶数输出。从 1～10 依次判断是否为偶数，如果是，则输出；如果不是，则继续下一次循环。**（实例位置：光盘\TM\Instance\03\3.5）**

实例代码如下：

```
<?php
	$num = 1;
	$str = "10 以内的偶数为：";
	while($num <= 10){
		if($num % 2 == 0){
			$str .= $num." ";
		}
		$num++;
	}
	echo $str;
?>
```

运行结果如图 3.14 所示。

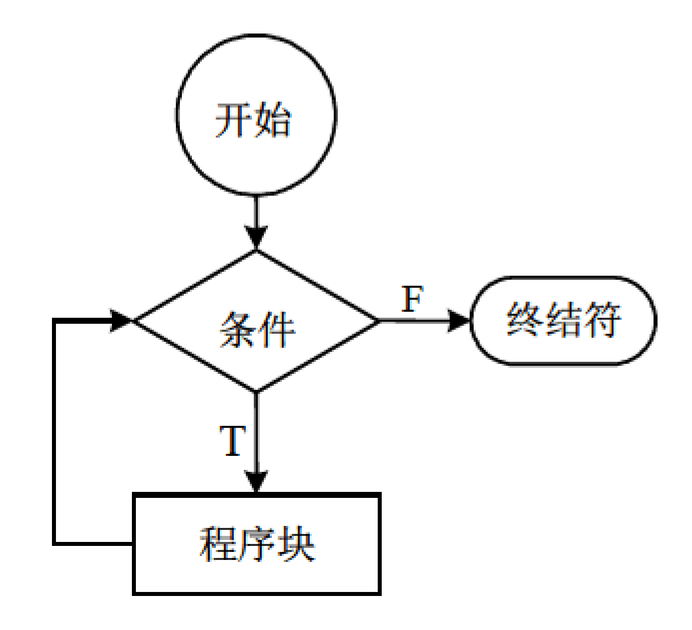


图 3.13　while 循环控制语句的操作流程

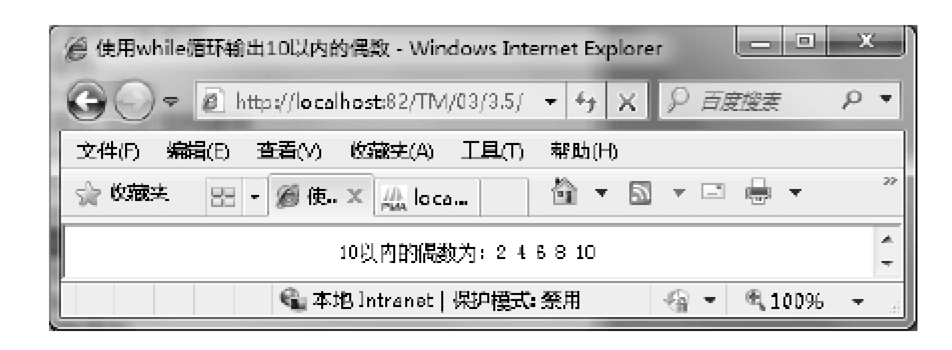


图 3.14　while 循环语句的应用

3.3.2　do…while 循环语句

视频讲解：光盘\TM\Video\第 3 章\do…while 循环语句.exe

while 语句还有另一种表示形式：do…while。do…while 循环语句和 while 循环语句非常相似，只是 do…while 循环语句在循环底部检测循环表达式，而不是在循环的顶部进行检测。

do…while 循环语句的语法格式如下：

```
do{
	statement
}while(expr);
```

do…while 语句先执行 statement 语句，然后再对表达式进行判断。如果表达式的值为 false，则跳出循环。因此应用 do…while 循环语句时该语句的循环体至少被执行一次。

do…while 语句的流程控制图如图 3.15 所示。

例 3.6　下面通过两个语句的运行对比来了解两者的不同。**（实例位置：光盘\TM\Instance\03\3.6）**

实例代码如下：

```
<?php
$num = 1;
while($num!=1){
	echo "不会看到";
}
do{
```

```
        echo "会看到";
}while($num!=1);
?>
```

结果为：会看到。

从上面的代码中可以看出两者的区别：do…while 要比 while 语句多循环一次。当 while 表达式的值为假时，while 循环直接跳出当前循环，而 do…while 语句则是先执行一遍程序块，然后再对表达式进行判断。

注意 do…while 语句结尾处的 while 语句括号后面有一个分号“;”，为了养成良好的编程习惯，建议读者在书写的过程中不要将其遗漏。

3.3.3 for 循环语句

视频讲解：光盘\TM\Video\第 3 章\for 循环语句.exe

for 循环是 PHP 中最复杂的循环结构。for 循环语句能够按照已知的循环次数进行循环操作，主要应用于多条件情况下的循环操作。如果在单一条件下使用 for 循环语句就有些不合适。这一点从该语句的语法中就可以看出，其条件的表达式有 3 个。它的语法格式为：

```
for (expr1; expr2; expr3){
        statement;
}
```

其中，expr1 为变量初始赋值；expr2 为循环条件，即在每次循环开始前求值。如果值为真，则执行 statement；否则，跳出循环，继续往下执行。expr3 为变量递增或递减，即每次循环后被执行。for 循环语句的流程图如图 3.16 所示。

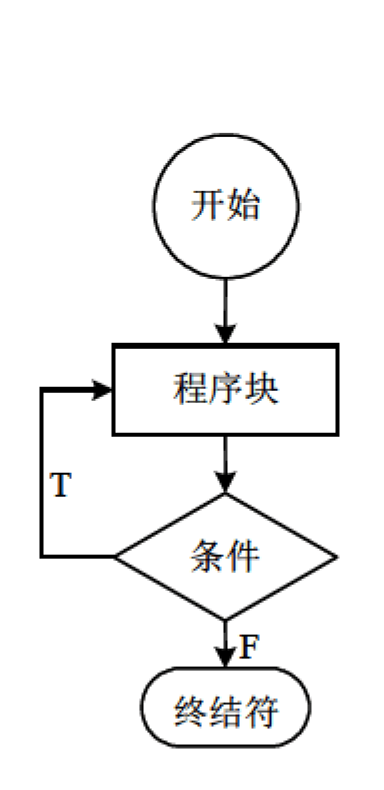


图 3.15　do…while 语句的控制流程图

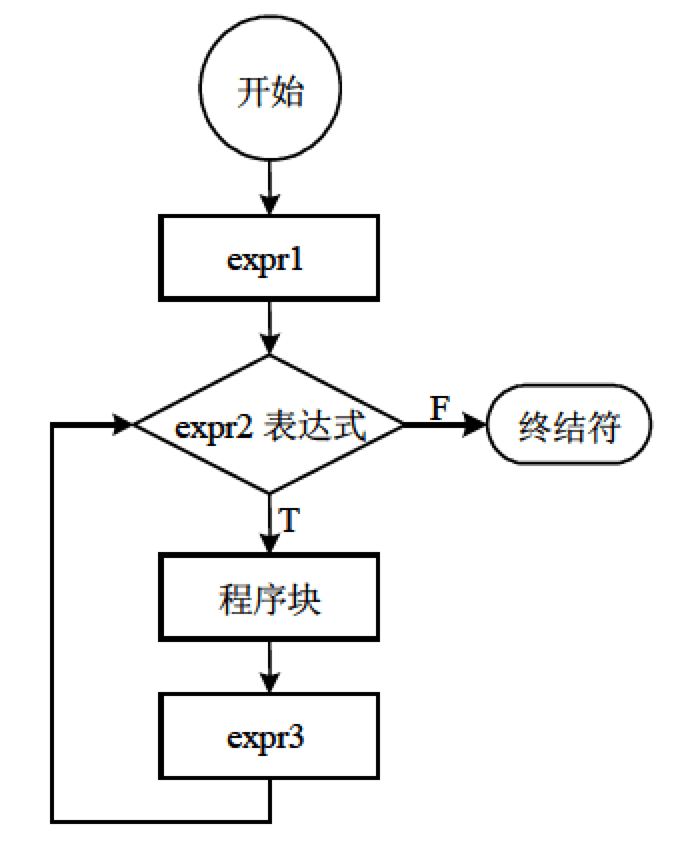


图 3.16　for 循环语句的流程图

例 3.7　应用 for 循环计算 50 的阶乘。（**实例位置：光盘\TM\Instance\03\3.7**）

代码如下：

```
<?php
        $sum = 1;                              //声明整型变量$sum
        for ($i = 1;$i <=50;$i++){
                $sum *= $i;                    //当$i≤50 时，执行该表达式
        }
        echo "50! = ".$sum;                    //输出该表达式
?>
```



结果为：50! = 3.04140932017E+64。

注意 在 for 语句中无论采用循环变量递增或递减的方式，有一个前提，就是一定要保证循环能够结束，无期限的循环（死循环）将导致程序的崩溃。在循环变量自增的例子中，循环的条件是$i≤50，由于在每次循环后$i 的值会加 1，当$i 的值大于 50 时，循环就结束了。

上面的代码采用了循环变量递增的方式。当然，也可以采用倒序的方式，以循环变量递减的方式编写程序。例如：

```
<?php
    $sum = 1;                                    //声明整型变量$sum
    for ($i = 50;$i >0;$i--){
        $sum *= $i;                              //当$i>0 时，执行该表达式
    }
    echo "50! = ".$sum;                          //输出该表达式
?>
```

结果为：50! = 3.04140932017E+64。

在循环变量自减的例子中，循环条件是$i>0，在每次循环后$i 的值会减 1，当$i≤0 时，循环就结束了。

3.3.4　foreach 循环

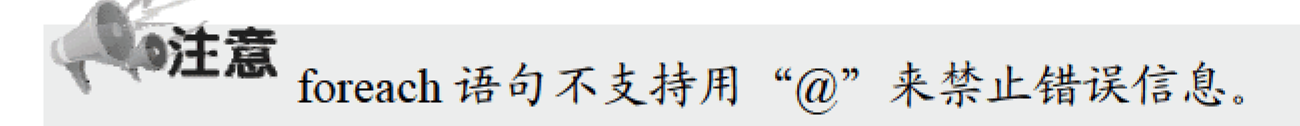
foreach 循环在 PHP 4 引进的，只能用于数组。在 PHP 5 中，又增加了对对象的支持。该语句的语法格式为：

```
foreach (array_expression as $value)
    statement
```

或

```
foreach (array_expression as $key => $value)
    statement
```

foreach 语句将遍历数组 array_expression，每次循环时，将当前数组中的值赋给$value（或是$key 和$value），同时，数组指针向后移动，直到遍历结束。当使用 foreach 语句时，数组指针将自动被重置，所以不需要手动设置指针位置。



注意 foreach 语句不支持用“@”来禁止错误信息。

例 3.8　foreach 语句的应用很广泛，其主要功能就是处理数组，下面就应用 foreach 语句来处理一个数组，实现输出购物车中商品的功能。这里假设将购物车中的商品存储于指定的数组中，然后通过 foreach 语句来输出购物车中的商品信息。**（实例位置：光盘\TM\Instance\03\3.8）**

程序代码如下：

```
<?php
$name = array("1"=>"iPhone5","2"=>"数码相机","3"=>"联想电脑","4"=>"天王表");
$price = array("1"=>"79999 元","2"=>"3000 元","3"=>"5600 元","4"=>"3600 元");
$counts = array("1"=>1,"2"=>1,"3"=>2,"4"=>1);
echo '<table width="580" border="1" cellpadding="1" cellspacing="1" bordercolor="#FFFFFF"
bgcolor="#c17e50">
        <tr>
          <td width="145" align="center" bgcolor="#FFFFFF"  class="STYLE1">商品名称</td>
          <td width="145" align="center" bgcolor="#FFFFFF"  class="STYLE1">价  格</td>
          <td width="145" align="center" bgcolor="#FFFFFF"  class="STYLE1">数量</td>
          <td width="145" align="center" bgcolor="#FFFFFF"  class="STYLE1">金额</td>
        </tr>';
```

```
foreach($name as $key=>$value){                         //以 book 数组做循环，输出键和值
        echo '<tr>
                <td height="25" align="center" bgcolor="#FFFFFF" class="STYLE2">'.$value.'</td>
                <td align="center" bgcolor="#FFFFFF" class="STYLE2">'.$price[$key].'</td>
                <td align="center" bgcolor="#FFFFFF" class="STYLE2">'.$counts[$key].'</td>
                <td align="center" bgcolor="#FFFFFF" class="STYLE2">'.$counts[$key]*$price[$key].'</td>
            </tr>';
}
echo '</table>';
?>
```

运行结果如图 3.17 所示。



图 3.17　应用 foreach 语句输出商品

注意 当试图使用 foreach 语句用于其他数据类型或者未初始化的变量时会产生错误。为了避免这个问题，最好使用 is_array()函数先来判断变量是否为数组类型，如果是，再进行其他操作。

3.4　跳转控制语句

跳转语句主要分为 3 个部分：break 语句、continue 语句和 return 语句。其中前两个跳转语句使用起来非常简单而且非常容易掌握，主要原因是它们都被应用在指定的环境中，如 for 循环语句中。return 语句在应用环境上较前两者相对单一。一般被用在自定义函数和面向对象的类中。

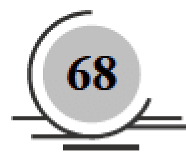
3.4.1　使用 break 语句跳出循环

break 关键字可以终止当前的循环，包括 while、do…while、for、foreach 和 switch 在内的所有控制语句。

例 3.9　将使用一个 while 循环，while 后面的条件表达式的值为 true，即为一个无限循环。在 while 程序块中将声明一个随机数变量$tmp，只有当生成的随机数等于 10 时，使用 break 语句跳出循环。**（实例位置：光盘\TM\Instance\03\3.9）**

代码如下：

```
<?php
while(true){                                            //使用 while 循环
      $tmp = rand(1,20);                                //声明一个随机数变量$tmp
      echo $tmp." ";                                    //输出随机数
```



```
    if($tmp == 10){                          //判断随机数是否等于 10
        echo "<p>变量等于 10，终止循环";
        break;                               //如果等于 10，使用 break 语句跳出循环
    }
}
?>
```

运行结果如图 3.18 所示。

break 语句不仅可以跳出当前的循环，还可以指定跳出几重循环。格式为：

```
break  n;
```

参数 n 指定要跳出的循环数量。break 关键字的流程图如图 3.19 所示。

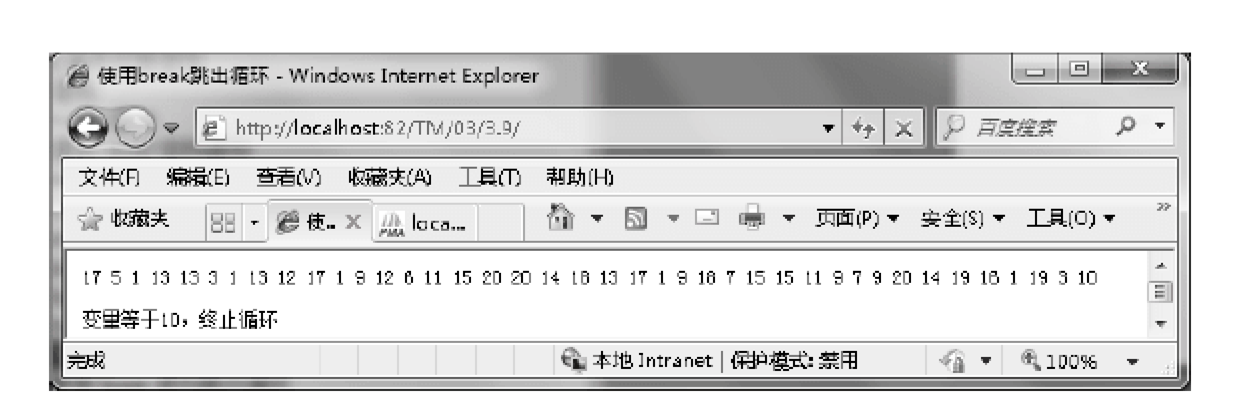


图 3.18　应用 break 跳转控制语句跳出循环

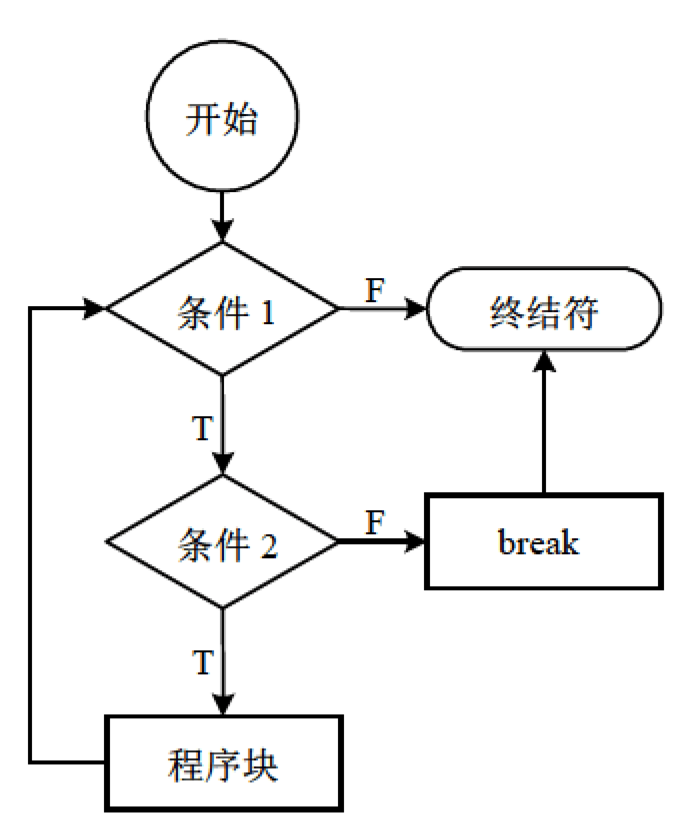


图 3.19　break 关键字的流程图

例 3.10　本例共有 3 层循环，最外层的 while 循环和中间层的 for 循环是无限循环，最里面并列两个 for 循环：程序首先执行第一个 for 循环，当变量$i 等于 6 时，跳出当前循环（一重循环），继续执行第二个 for 循环，当第二个 for 循环中的变量$j 等于 10 时，将直接跳出最外层循环。**（实例位置：光盘\TM\Instance\03\3.10）**

实例代码如下：

```
<?php
while(true){
    for(;;){
        for($i=0;$i<=10;$i++){
            echo $i." ";
            if($i == 6){
                echo "<p>变量\$i 等于 6，跳出一重循环。<p>";
                break 1;
            }
        }
        for($j = 0; $j < 20; $j++){
            echo $j." ";
            if($j == 10){
                echo "<p>变量\$j 等于 10，跳出最外重循环。";
                break 3;
            }
        }
    }
    echo "这句话不会被执行。";
}
?>
```

运行结果如图 3.20 所示。

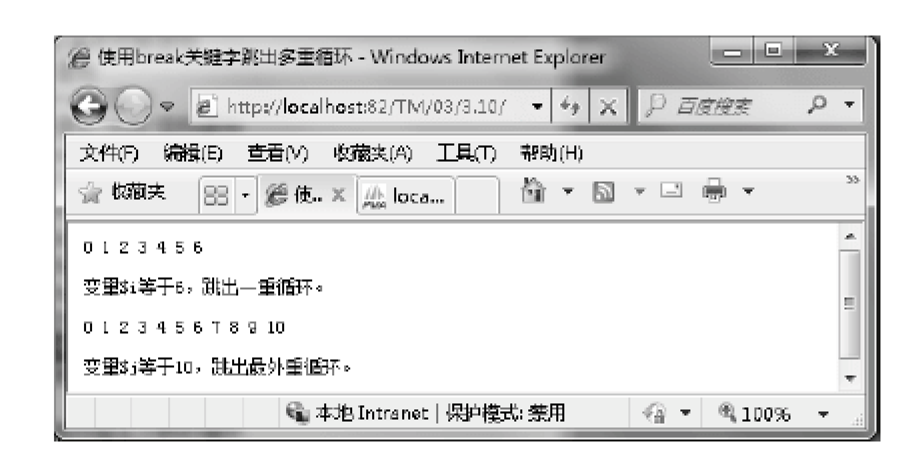


图 3.20　使用 break 关键字跳出多重循环

3.4.2　使用 continue 语句跳出循环

continue 跳转语句的作用没有 break 那么强大，只能终止本次循环，而进入到下一次循环中。在执行 continue 语句后，程序将结束本次循环的执行，并开始下一轮循环的执行操作。continue 也可以指定跳出几重循环。continue 跳转语句的流程图如图 3.21 所示。

例 3.11　本例将使用 for 循环输出数组变量。如果变量的数组下标为偶数，则只输出一个空行；如果是奇数，才继续输出。在最里面的循环中，判断当前数组下标是否等于$i，如果不相等，输出数组变量，否则跳到最外重循环。**（实例位置：光盘\TM\Instance\03\3.11）**

代码如下：

```
<?php
    $arr = array("PHP 程序开发实战宝典","PHP 求职宝典","PHP 典型模块","PHP 项目开发全程实录","PHP 开发实战 1200 例","PHP 网络编程自学手册");            //声明一个数组变量$arr
    for($i = 0; $i < 6; $i++){                  //使用 for 循环
        echo "<br>";
        if($i % 2 == 0){                        //如果$i 的值为偶数，则跳出本次循环
            continue;
        }
        for(;;){                                //无限循环
            for($j = 0; $j < count($arr); $j++){    //再次使用 for 循环输出数组变量
                if($j == $i){                   //如果当前输出的数组下标等于$i
                    continue 3;                 //跳出最外重循环
                }else{
                    echo $arr[$j]."\t | ";      //输出表达式
                }
            }
        }
        echo "不会看到!";
    }
?>
```

运行结果如图 3.22 所示。

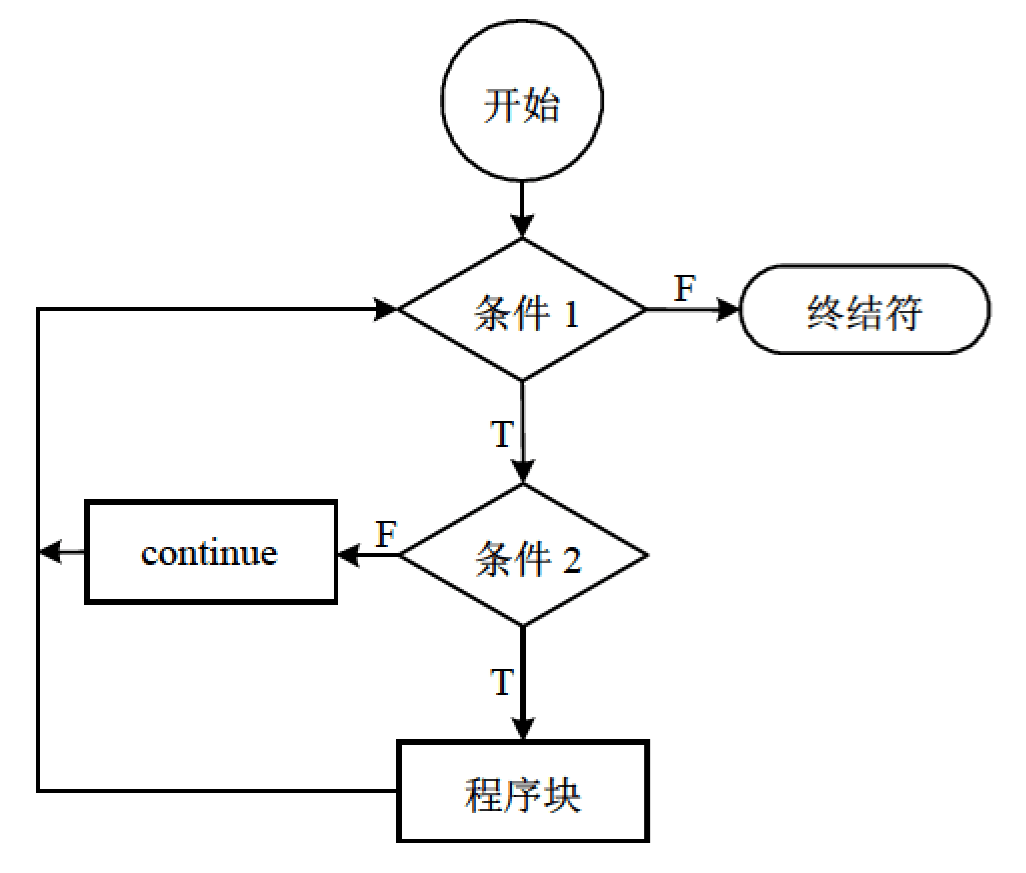


图 3.21　continue 跳转语句的流程图

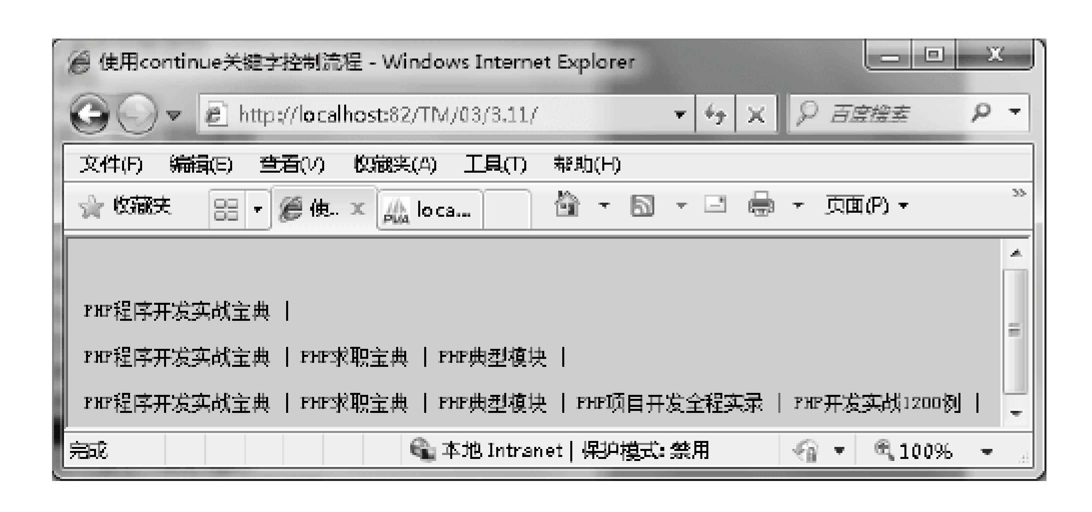
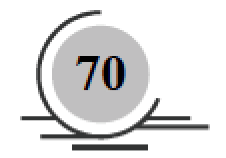


图 3.22　使用 continue 语句控制流程的应用



3.5 实　　战

3.5.1 执行指定次数的循环

例 3.12 利用 break 语句实现执行指定次数的循环。（**实例位置：光盘\TM\Instance\03\3.12**）

代码如下：

```
<table width="761" border="0" align="center" cellpadding="0" cellspacing="0" bordercolor="#FEFEFE" bgcolor=
"#FFFFFF">
  <form action=""  method="post" name="form1" id="form1">
    <tr>
      <td width="761" align="center" bgcolor="#F9F8EF"><table width="749" border="0" align="center"
cellpadding="0"  cellspacing="0"  style="BORDER-COLLAPSE: collapse">
        <tr>
          <td width="749" height="57" background="images/a_03.jpg">  </td>
        </tr>
        <tr>
          <td  height="36"  colspan="3"  align="left"  background="images/a_05.jpg"  bgcolor="#F9F8EF"
scope="col">姓　名：
            <input  name="user_name" id="user_name" value=" 匿名" maxlength="64" type="text" />
              <span
            style="COLOR: #ff0000">*</span></td>
        </tr>
        <tr>
          <td height="36" colspan="3" align="left" background="images/a_05.jpg" bgcolor="#F9F8EF">标　题：
            <input maxlength="64" size="30" name="title"  type="text"/>
              <span style="COLOR: #ff0000">*</span></td>
        </tr>
        <tr>
          <td          height="126"          colspan="3"          align="left"          background="images/a_05.jpg"
bgcolor="#F9F8EF">         内  容：
            <textarea  name="content"  cols="60"  rows="8"  id="content"  style="background:url
(./images/ mrbccd.gif)"></textarea>
              <span style="COLOR: #ff0000">*</span></td>
        </tr>
        <tr>
          <td height="40" colspan="3" align="center" background="images/a_05.jpg" bgcolor="#F9F8EF">
     <input name="submit" type="submit" class="btn1" id="submit" value="签写留言"/>
     <input name="reset" type="reset" class="btn1" value="清除留言" />
          </td>
        </tr>
      </table>
     </td>
    </tr>
    <tr>
      <td width="761" align="center" bgcolor="#F9F8EF">
         <table width="749" border="0" align="center" cellpadding="0" cellspacing="0">
            <tr>
```

```
                <td width="703" height="31" align="center" background="images/a_07.jpg"> </td>
              </tr>
          </table></td>
      </tr>
  </form>
</table>
```

当用户发表留言信息后，提交留言信息时，通过正则表达式 preg_match()函数将留言信息与存储在数组中的敏感词进行对比，检验用户提交的留言信息中是否含有敏感词。如果留言信息中含有敏感词，那么将弹出提示信息，否则留言信息发布成功。实现敏感词过滤的完整代码如下：

```
<?php
    if($_POST[sub1]){
        for($a = 0;$a < 10000000000000;$a++){
            if($a >= $_POST[text]){
                break;
            }else{
                echo "已经输出".($a+1)."次循环<br>";
            }
        }
    }
?>
```

运行结果如图 3.23 所示。

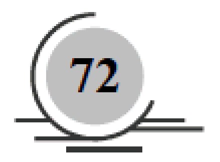
3.5.2 跳过数据输出中指定的记录

例 3.13 应用 continue 语句跳过指定的循环，输出指定循环以外的数据。（**实例位置：光盘\TM\Instance\03\ 3.13**）

代码如下：

```
<?php
    $array = array("PHP 典型模块","PHP 开发实战宝典","JAVA 开发实战宝典","PHP 网络编程自学手册");
?>
<form action="" method="post">
    <textarea name="te" cols="20" rows="6">0、<?php echo $array[0]?>        1、<?php echo $array[1]?>
2、<?php echo $array[2]?>                          3、<?php echo $array[3]?>
    </textarea><br>
    <input type="text" name="text" value="输入要跳过记录的编号"size="20" onFocus="this.value=''">
    <input type="submit" name="sub1" value="跳过">
</form>
<?php
    if($_POST[sub1]){
        for($a = 0;$a < count($array);$a++){
            if($a == $_POST[text]){
                continue;
            }else{
                echo '《';
                echo $array[$a];
                echo '》';
            }
        }
    }
?>
```

运行结果如图 3.24 所示。



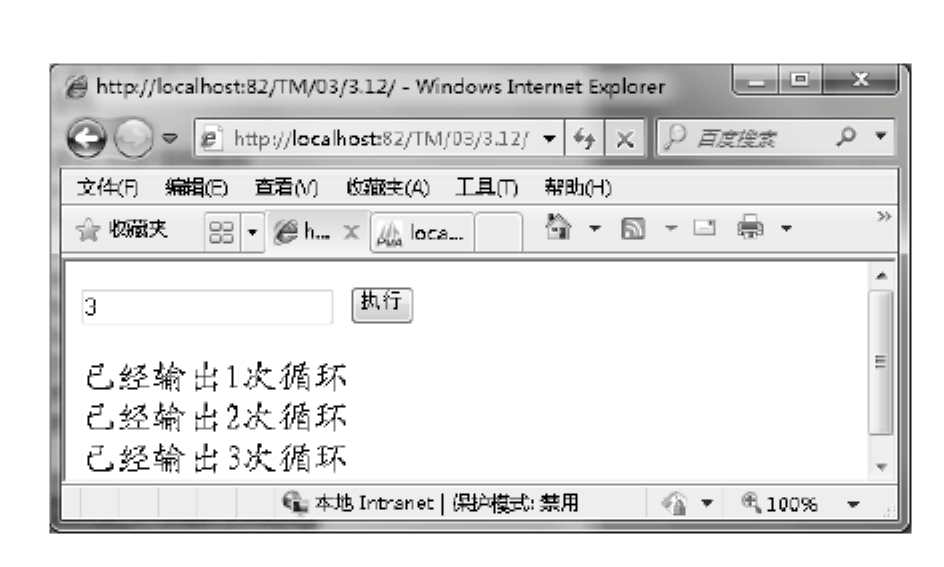


图 3.23　执行指定次数的循环

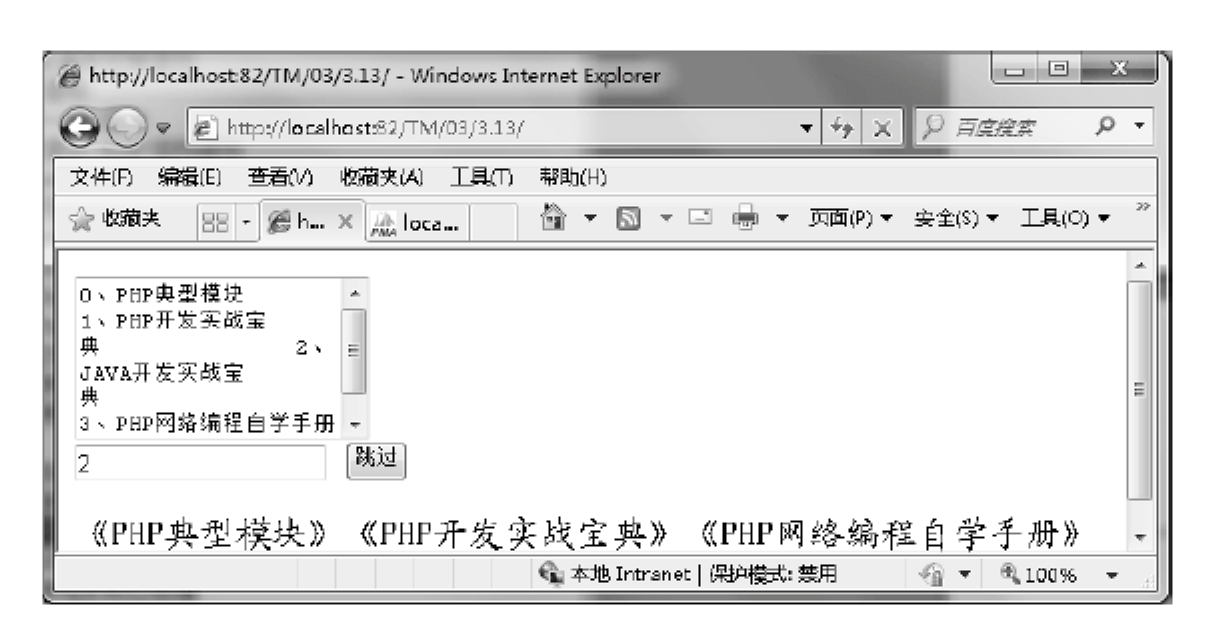


图 3.24　跳过数据输出中指定的记录

3.5.3　控制页面中数据的输出数量

例 3.14　运用 break 语句跳出指定的几重循环。（**实例位置：光盘\TM\Instance\03\3.14**）

代码如下：

```
<?php
	for($a = 0;$a < 2;$a++){
		for($b = 0;$b < 2;$b++){
			for($c = 0;$c < 4;$c++){
				$rand = rand(1,4);
				echo "<img src='image/$rand.gif'>";
			}
		}
	}
	echo "<hr style='color:blue;'>";
	for($a = 0;$a < 2;$a++){
		for($b = 0;$b < 2;$b++){
			for($c = 0;$c < 4;$c++){
				$rand = rand(1,4);
				if($rand != 2){
					echo "<img src='image/$rand.gif'>";
				}else{
					break 3;
				}
			}
		}
	}
?>
```

运行本例，单击栏目导航条中的超链接，将在主显示区中输出对应的项目内容，如图 3.25 所示。



图 3.25　控制页面中数据的输出数量

3.5.4　动态改变页面中单元格的背景颜色

例 3.15　通过使用 continue 语句实现动态改变页面中单元格的背景颜色。（**实例位置：光盘\TM\Instance\03\3.15**）

代码如下：

```
<?php
		for($b = 0;$b < 10;$b++){
			if($b <= 5){
```

```
                    $rand.= dechex(rand(0,15));
               }else{
                    continue;
               }
          }
?>
<table height="55" bgcolor="#<?php echo $rand;?>">
   <tr><td width="156"></td>
</tr></table>
```

运行结果如图 3.26 所示。

3.5.5 使用 for 循环动态创建表格

例 3.16 通过使用 for 循环动态创建表格。**（实例位置：光盘\TM\Instance\03\3.16）**

代码如下：

```
<?php
     for($a = 0;$a < 5;$a++){
          echo "<table border='1'bordercolor='#000099' bordercolorlight='#000099' cellspacing='0'>";
          echo "<tr>";
          for($b = 0;$b < 7;$b++){
               echo "<td>";
               echo $b;
               echo "</td>";
          }
          echo "</tr>";
          echo "</table>";
     }
?>
```

运行结果如图 3.27 所示。

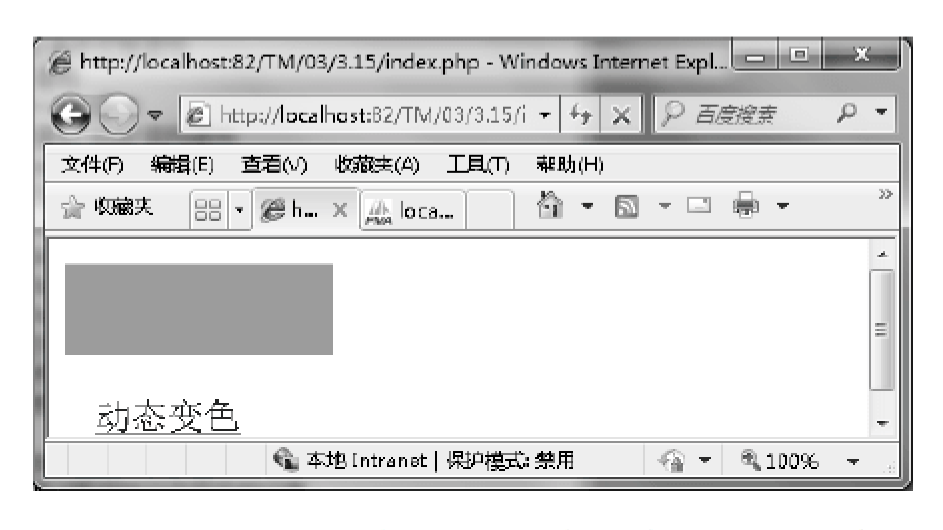


图 3.26 动态改变页面中单元格的背景颜色



图 3.27 表格的动态创建

3.6 小 结

本章主要讲述的是流程控制语句的知识，并且对算法和程序的控制结构进行了介绍。重点掌握 3 种流程控制语句——条件控制语句、循环控制语句和跳转控制语句。在掌握流程控制语句的基础上，对程序的算法和控制结构应该有所了解。读者通过对本章的学习能够从宏观的角度去认识 PHP 语言，从整体上形成一个开发的思路，逐渐形成一种属于自己的编程思想和编程方法。



3.7　学习成果检验

1．应用 switch…case 分支控制语句输出本年度当前月份的活动安排。**（实例位置：光盘\TM\Instance\03\3.17）**

2．在开发动态网站时应用 do…while 循环语句输出企业公告信息。**（实例位置：光盘\TM\Instance\03\3.18）**

3．使用循环语句，输出任意一个二维数组。**（实例位置：光盘\TM\Instance\03\3.19）**

4．使用循环控制语句，输出杨辉三角。**（实例位置：光盘\TM\Instance\03\3.20）**

第 4 章

字符串操作与正则表达式

（视频讲解：92 分钟）

在 Web 编程中，字符串总是会被大量地生成和处理。正确地使用和处理字符串，对于 PHP 程序员来说越来越重要。并且在新技术层出不穷的今天，让人难忘的、能称得上是伟大技术的寥寥无几，但其中一定会有正则表达式，然而，最容易被人忽略和遗忘的也是正则表达式。本章从最简单的字符串定义开始讲解正则表达式的知识，希望广大读者通过本章的学习，了解和掌握 PHP 字符串和正则表达式。

通过阅读本章内容，你可以：

- 掌握单引号和双引号的区别
- 掌握字符串的连接方法
- 熟悉去除字符串中的空格的方法
- 掌握查找和替换字符串技术
- 了解正则表达式的发展及相关概念
- 了解 PHP 中的 POSIX 函数
- 了解 PHP 中的 PCRE 函数
- 掌握正则表达式的应用

4.1　了解字符串

字符串是由零个或多个字符组成的有限序列。字符包含以下几种类型：

☑　数字类型，如 1、2、3 等。

☑　字母类型，如 a、b、c、d 等。

☑　特殊字符，如$、%、#、^、&等。

☑　不可见字符，如\t（Tab 字符）、\n（换行符）、\r（回车符）等。

其中，不可见字符是比较特殊的一组字符，是用来控制字符串格式化输出的，在浏览器上不可见，只能看到字符串输出的结果。

例如：

```
<?php
    echo "PHP \t SQL Server \n MySQL \r Java";      //输出字符串
?>
```

结果为：PHP SQL Server
MySQL
Java。

注意　应用本例在 IE 浏览器上看不到不可见字符的输出结果，需要通过选择 IE 浏览器菜单中的“查看源文件”命令来查看执行不可见字符串的实际输出结果。

4.2　单引号与双引号

视频讲解：光盘\TM\Video\第 4 章\单引号与双引号.exe

字符串通常以串的整体作为操作对象，一般用双引号或单引号标识一个字符串。单引号串和双引号串在使用上有一定区别。

下面分别使用单引号和双引号来定义一个字符串。

```
<?php
$str = 6;
echo "str is $str";                    //输出结果：str is 6
echo 'str is $str';                    //输出结果：str is $str
echo "str is $str\n";                  //输出结果：str is 6（同时换行）
echo 'str is $str\n';                  //输出结果：str is $str\n
?>
```

从以上代码中可以看出，双引号中的内容是经过 PHP 的语法分析器解析过的，任何变量在双引号中都会被转换为它的值进行输出显示；而单引号的内容是“所见即所得”的，无论有无变量，都被当作普通字符串进行原样输出。

然而，对于定义的普通字符串却显示不出两者之间的区别。例如：

```
<?php
$str1 = "I Like eat";                  //应用双引号定义一个字符串
$str2 = 'I Like eat';                  //应用单引号定义一个字符串
echo $str1;                            //输出双引号中的字符串
```

```
echo $str2;                                   //输出单引号中的字符串
?>
```

结果为：I Like eat

I Like eat。

技巧 单引号串和双引号串在 PHP 中的处理是不相同的。双引号串中的内容可以被解释而且替换，而单引号串中的内容被作为普通字符进行处理。

4.3 定　界　符

定界符（<<<）是从 PHP 4.0 开始支持的。定界符用于定义格式化的大文本，格式化是指文本中的格式将被保留，所以文本中不需要使用转义字符。在使用时后接一个标识符，然后是格式化文本（即字符串），最后是同样的标识符结束字符串。

定界符格式如下：

```
<<<str
     格式化文本
str;
```

其中，符号“<<<”是关键字，必须使用；str 为用户自定义的标识符，用于定义文本的起始标识符和结束标识符，前后的标识符名称必须完全相同。

结束标识符必须从行的第一列开始。而且也必须遵循 PHP 中其他标签的命名规则：只能包含字母、数字、下划线，而且必须以下划线或非数字字符开始。

例 4.1　应用定界符输出变量中的值，可以看到，它和双引号没什么区别，包含的变量也被替换成实际数值。**（实例位置：光盘\TM\Instance\04\4.1）**

代码如下：

```
<?php
$str="敬请大家";
echo <<<strmark
<font color="#FF0099"> $str 关注明日图书网：www.mingribook.com </font>
strmark;
?>
```

在上面的代码中，值得注意的是，在定界符内不允许添加注释，否则程序将运行出错。结束标识符所在的行不能包含任何其他字符，而且不能被缩进，在标识符分号前后不能有任何空白字符或制表符。如果破坏了这条规则，则程序不会被视为结束标识符，PHP 将继续寻找下去。如果在这种情况下找不到合适的结束标识符，将会导致一个在脚本最后一行出现的语法错误。

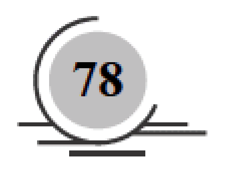
运行结果如图 4.1 所示。

技巧 定界符中的字符串支持单引号、双引号，无须转义，并支持字符变量替换。

例 4.2　应用定界符输出一个用户登录的表单。**（实例位置：光盘\TM\Instance\04\4.2）**

代码如下：

```
<?php
$str=<<<user
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
xhtml1- transitional.dtd">
```



```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>用户登录</title>
<link rel="stylesheet" type="text/css" href="style.css" />
<body>
<table width="250" height="100" border="0" align="center" cellpadding="0" cellspacing="0">
  <tr>
    <td bgcolor="#666666"><table width="250" height="100" border="0" cellpadding="0" cellspacing="1">
     <form name="form1" method="post" action="index.php">
       <tr>
         <td height="25" colspan="2" bgcolor="#DDAACC"><div align="center" >用户登录</div></td>
       </tr>
       <tr>
         <td width="60" height="25" bgcolor="#FFFFFF"><div align="right">用户名：</div></td>
         <td width="187" bgcolor="#FFFFFF"> <input type="text" name="ip" size="18" 
class="inputcss"></td>
       </tr>
       <tr>
         <td height="25" bgcolor="#FFFFFF"><div align="right">密码：</div></td>
         <td height="25" bgcolor="#FFFFFF"> <input type="text" name="ipNum" size="18" 
class="inputcss"> </td>
       </tr>
       <tr>
         <td height="25" colspan="2" bgcolor="#FFFFFF"><div align="center"><input name="submit" 
type="submit" class="buttoncss" value="登录" />
</div></td>
       </tr>
      </form>
    </table></td>
  </tr>
</table>
</body>
</html>
user;
echo $str;
?>
```

注意 在上面的代码中，结束标识符 user 必须单独另起一行，并且不允许有空白字符。如果在标识符前后有其他符号、字符或注释，则会发生错误。

运行结果如图 4.2 所示。



图 4.1 使用定界符定义字符串

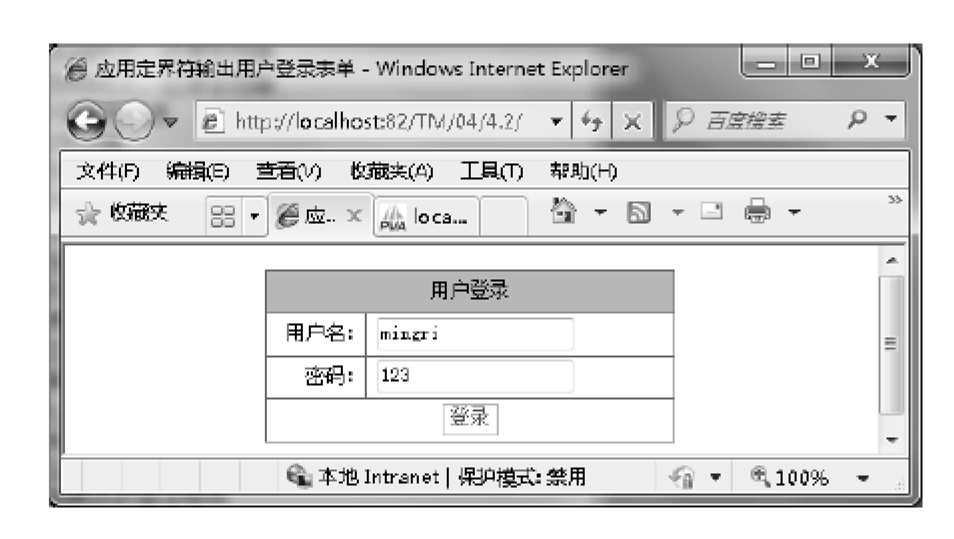


图 4.2 使用定界符输出用户登录表单

4.4 连接字符串

视频讲解：光盘\TM\Video\第 4 章\连接字符串.exe

字符串可以用“.”（点）字符串连接符连接，它可以把两个或两个以上的字符串连接成一个新的字符串。注意这里不能用“+”（加）运算符。

字符串连接有两种形式，第一种是连接运算符“.”；第二种是连接赋值运算符“.=”。

连接运算符“.”返回其左右参数连接后的字符串。例如：

```
<?php
$name="明日图书网：";
$url = $name."www.mrbccd.com";
echo $url;
?>
```

结果为：明日图书网：www.mrbccd.com。

连接赋值运算符“.=”将右边参数附加到左边的参数后连接成一个新的字符串。例如：

```
<?php
$url="明日图书网：";
$url .= "www.mrbccd.com ";
echo   $url;
?>
```

结果为：明日图书网：www.mrbccd.com。

4.5 转义、还原字符串

在 PHP 编程的过程中，经常会遇到这样的问题，将数据插入到数据库时可能引起一些问题，出现错误或者乱码等，因为数据库将传入的数据的字符解释成了控制符。针对这种问题，就需要使用一种标记或者是转义这些特殊的字符。

因此，在 PHP 语言中提供了专门处理这些问题的技术，转义和还原字符串。方法有两种：一种是手动转义、还原字符串数据，另一种是自动转义、还原字符串数据。下面分别对这两种方法进行详细讲解。

4.5.1 手动转义、还原字符串

字符串可以用单引号（'）、双引号（""）、定界符（<<<）3 种方式定义。而指定一个简单字符串的最简单的方法是用单引号（'）括起来。当使用字符串时，很可能在该串中存在这几种符号与 PHP 脚本混淆的字符，因此必须要做转义语句，即在它的前面使用转义符号（\）。

“\”是一个转义符，紧跟在“\”后面的第一个字符将变为没有意义或特殊意义。例如，“'”是字符串的定界符，写成“\'”这样时就使它失去了定界符的意义，变为普通的单引号“'”。读者可以通过“echo '\'';”输出一个单引号“'”，同时转义字符“\”不会显示。

技巧 如果要在字符串中表示单引号，则需要用反斜杠“\”进行转义。例如，要表示字符串“I'm”，则需要写成“I\'m”。

例 4.3　使用转义字符（\）对字符串进行转义。**（实例位置：光盘\TM\Instance\04\4.3）**

代码如下：

```
<?php
    echo ' select * from book where bookname = \'PHP 开发实战宝典\' ';
?>
```

结果为：select * from book where bookname = 'PHP 开发实战宝典'。

技巧 对于简单的字符串，建议采用手动方法进行字符串转义，而对于数据量较大的字符串，建议采用自动转义函数实现字符串的转义。

4.5.2　自动转义、还原字符串

自动转义、还原字符串数据可以应用 PHP 提供的 addslashes()函数和 stripslashes()函数实现。

☑　addslashes()函数

addslashes()函数用来给字符串 str 加入反斜杠（\），对指定字符串中的字符进行转义，该函数可以转义的字符包括单引号（'）、双引号（"）、反斜杠（\）、NULL 字符（0）。该函数比较常用的地方就是在生成 SQL 语句时，对 SQL 语句中的部分字符进行转义。

语法：

```
string addslashes ( string str)
```

参数 str 为将要被操作的字符串。

☑　stripslashes()函数

stripslashes()函数用来将应用 addslashes()函数转义后的字符串 str 返回原样。

语法：

```
string stripslashes(string str);
```

参数 str 为将要被操作的字符串。

例 4.4　使用自动转义字符 addslashes()函数对字符串进行转义，然后应用 stripslashes()函数进行还原。**（实例位置：光盘\TM\Instance\04\4.4）**

代码如下：

```
<?php
 $str = "select * from book where bookname = 'PHP 开发实战宝典'";
 $a = addslashes($str);                              //对字符串中的特殊字符进行转义
 echo $a."<br>";                                     //输出转义后的字符
 $b = stripslashes($a);                              //对转义后的字符进行还原
 echo $b."<br>";                                     //将字符原义输出
?>
```

运行结果如图 4.3 所示。

技巧 所有数据在插入数据库前，有必要应用 addslashes()函数进行字符串转义，以免特殊字符未经转义在插入数据库时出现错误。另外，对于应用 addslashes()函数实现的自动转义字符串可以应用 stripslashes()函数进行还原，但数据在插入数据库前必须再次进行转义。

以上两个函数实现了对指定字符串进行自动转义和还原。除了上面介绍的方法外，还可以对要转义、还原的字符串进行一定范围的限制，PHP 通过应用 addcslashes()函数和 stripcslashes()函数实现对指定范围内

的字符串进行自动转义、还原。

☑ addcslashes()函数

实现对指定字符串中的字符进行转义，即在指定的字符 charlist 前加上反斜杠（\）。通过该函数可以将要添加到数据库中的字符串进行转义，从而避免出现乱码等问题。

语法：

```
string addcslashes ( string str, string charlist)
```

参数 str 为将要被操作的字符串；参数 charlist 指定在字符串中哪些字符前加上反斜杠（\），如果参数 charlist 中包含有“\n”、“\r”等字符，将以 C 语言风格转换，而其他非字母数字且 ASCII 码低于 32 或高于 126 的字符均转换成以八进制表示。


注意 在定义参数 charlist 的范围时，需要明确开始和结束范围内的字符串。

☑ stripcslashes()函数

stripcslashes()函数用来将应用 addcslashes()函数转义的字符串 str 还原。

语法：

```
string stripcslashes ( string str)
```

参数 str 为将要被操作的字符串。

例 4.5 应用 addcslashes()函数对字符串“明日科技”进行转义，应用 stripcslashes()函数对转义的字符串进行还原。（**实例位置：光盘\TM\Instance\04\4.5**）

代码如下：

```
<?php
$a="明日科技";                              //对指定范围内的字符进行转义
$b=addcslashes($a,"明日科技");               //转义指定的字符串
echo "转义字符串：".$b;                      //输出转义后的字符串
echo "<br>";                                 //执行换行
$c=stripcslashes($b);                        //对转义的字符串进行还原
echo "还原字符串：".$c;                      //输出还原后的转义字符串
?>
```

运行结果如图 4.4 所示。

图 4.3　自动转义和还原字符串

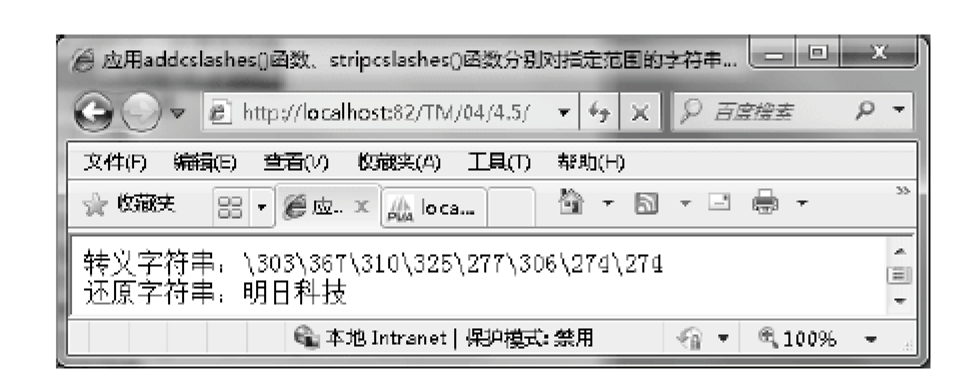


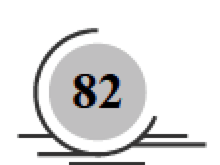
图 4.4　对指定范围的字符串进行转义和还原

技巧 缓存文件中，一般对缓存数据的值采用 addcslashes()函数进行指定范围的转义。

4.6　获取字符串长度

视频讲解：光盘\TM\Video\第 4 章\获取字符串长度.exe

获取字符串长度主要通过 strlen()函数实现，下面重点讲解 strlen()函数的语法及其应用。



strlen()函数主要用于获取指定字符串 str 的长度。

语法：

```
int strlen(string str)
```

例 4.6　应用 strlen()函数来获取指定字符串的长度。（**实例位置：光盘\TM\Instance\04\4.6**）

代码如下：

```
<?php
echo strlen("明日科技图书网:www.mingribok.com");              //输出指定字符串长度
?>
```

结果为：32。



说明　汉字占两个字符，数字、英文、小数点、下划线和空格各占一个字符。

strlen()函数在获取字符串长度的同时，也可以用来检测字符串的长度。

例 4.7　应用 strlen()函数对提交的用户密码的长度进行检测，如果其长度小于 4 位，则弹出提示信息。（**实例位置：光盘\TM\Instance\04\4.7**）

（1）利用开发工具（如 Dreamweaver），新建一个 PHP 动态页，存储为 index.php。

（2）添加一个表单，将表单的 action 属性设置为 index_ok.php。

```
<form name="form1" method="post" action="index_ok.php">
    <tr>
        <td height="33"></td>
        <td align="center"><span class="style1">用户名</span>:</td>
        <td>
        <input name="user" type="text" id="user" size="15">
        </td>
        <td align="center">
        <input  name="imageField"  type="image"  src="images/btn_dl.jpg"  width="50"  height="20"
border="0">
        </td>
        <td> </td>
    </tr>
    <tr>
       <td height="27"> </td>
       <td align="center"><span class="style1">密码</span>:</td>
       <td>
           <input name="pwd" type="password" id="pass" size="15">
       </td>
       <td colspan="2" align="left"><span class="STYLE3">* 密码长度不能少于 4 位</span></td>
      </tr>
</form>
```

（3）应用 HTML 标记设计页面，添加一个“用户名”文本框，命名为 user；添加一个“密码”文本框，命名为 pwd；添加一个图像域，指定源文件位置为 images/btn_dl.jpg。

（4）再新建一个 PHP 动态页，存储为 index_ok.php。通过 POST 方法（关于 POST 方法将在后面章节中进行详细讲解）接收用户输入的用户密码的值。应用 strlen()函数获取用户提交密码的长度，应用 if 条件控制语句判断密码长度是否小于 4，并给出相应的提示信息。代码如下：

```
<?php
if(strlen($_POST["pwd"])<4){                          //检测用户密码的长度小于 4，弹出警告信息
     echo "<script>alert('用户密码的长度不得少于 4 位!请重新输入'); history.back();</script>";
}
else{                                                 //用户密码大于等于 4 位，则弹出该提示信息
```

```
    echo "用户信息输入合法！";
}
?>
```

（5）在 IE 浏览器中输入地址，按 Enter 键，运行结果如图 4.5 所示。



图 4.5　应用 strlen()函数检测字符串的长度

4.7　截取字符串

视频讲解：光盘\TM\Video\第 4 章\截取字符串.exe

在 PHP 中，有一项非常重要的技术，就是截取指定字符串中指定长度的字符。PHP 对字符串截取可以通过 PHP 的预定义函数 substr()实现。本节重点介绍字符串的截取技术。

substr()函数从字符串中按照指定位置截取一定长度的字符。通过该函数可以获取某个固定格式字符串中的一部分，如果使用一个正数作为子串起点来调用这个函数，将得到从起点到字符串结束的这个字符串；如果使用一个负数作为子串起点来调用，将得到一个原字符串尾部的子串，字符个数等于给定负数的绝对值。

语法：

```
string substr ( string str, int start [, int length])
```

参数说明如表 4.1 所示。

表 4.1　substr 函数的参数说明

参　　数	说　　明
str	指定字符串对象
start	指定开始截取字符串的位置，如果参数 start 为负数，则从字符串的末尾开始截取
length	指定截取字符的个数，如果 length 为负数，则表示截取到倒数第 length 个字符



注意　本函数中参数 start 的指定位置是从 0 开始计算的，即字符串中的第一个字符表示为 0。

例 4.8　应用 substr()函数截取字符串中指定长度的字符。（**实例位置：光盘\TM\Instance\04\4.8**）

代码如下：

```
<?php
echo substr("www.mingribook.com",0);                //从第 0 个字符开始截取
echo "<br>";
echo substr("www.Mingribook. ",3,8);                //从第 3 个字符开始连续截取 8 个字符
echo "<br>";
echo substr("www.Mingribook.com ",-3,3);            //从倒数第 3 个字符开始截取 3 个字符
```

```
echo "<br>";
echo substr("www.Mingribook.com ",0,-2);                //从第一个字符开始截取，截取到倒数第 2 个字符
?>
```

结果为：www.mingribook.com
.mingrib
om
www.mingribook.co。

例 4.9　下面应用 substr()函数截取超长文本的部分字符串，剩余的部分用“…”代替。**（实例位置：光盘\TM\Instance\04\4.9）**

代码如下：

```
<?php
$str="明日科技有限公司是一家知名企业的软件公司，公司主要经营图书开发词典，各种图书。";
if(strlen($str)>40){                              //如果文本的字符串长度大于 40
    echo substr($str,0,40)."…";                   //输出文本的前 40 个字符串，然后输出省略号
}
else{                                             //如果文本的字符串长度小于 40
    echo $str;                                    //直接输出文本
}
?>
```

结果为：明日科技有限公司是一家知名企业的软件公司…。

4.8　比较字符串

在 PHP 中，对字符串之间进行比较的方法有很多种：第一种应用 strcmp()和 strcasecmp()函数按照字节进行比较；第二种应用 strnatcmp()函数按照自然排序法进行比较；第三种应用 strncmp()函数指定从源字符串的位置开始比较。下面分别对这 3 种方法进行讲解。

4.8.1　按字节比较

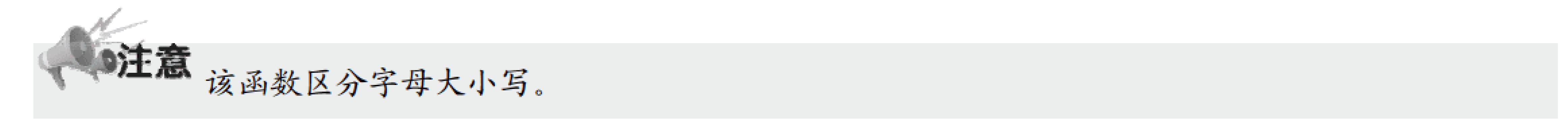
按字节进行字符串比较的方法有两种，分别为使用 strcmp()函数和 strcasecmp()函数进行比较。这两种函数的区别是 strcmp()函数区分字符的大小写，而 strcasecmp()函数不区分字符的大小写。这两个函数的实现方法基本相同，这里只介绍 strcmp()函数。

strcmp()函数用来对两个字符串进行比较。

语法：

```
int strcmp ( string str1, string str2))
```

参数 str1 和参数 str2 指定要比较的两个字符串。如果相等则返回 0；如果参数 str1 大于参数 str2，则返回 1；如果参数 str1 小于参数 str2，则返回-1。



该函数区分字母大小写。

例 4.10　应用 strcmp()和 strcasecmp()函数分别对两个字符串按字节进行比较。**（实例位置：光盘\TM\Instance\04\ 4.10）**

代码如下：

```
<?php
$str1="明日科技!";                                //定义字符串常量
$str2="明日科技!";                                //定义字符串常量
$str3="mrsoft";                                   //定义字符串常量
$str4="MRSOFT";                                   //定义字符串常量
echo strcmp($str1,$str2);                         //这两个字符串相等
echo strcmp($str3,$str4);                         //注意该函数区分大小写
echo strcasecmp($str3,$str4);                     //该函数不区分字母大小写
?>
```

结果为：0 1 0。

技巧 在 PHP 中，对字符串之间进行比较的应用也是非常广泛的。例如，应用 strcmp()函数来比较在用户登录系统中输入的用户和密码是否正确。如果在验证用户名和密码时不使用此函数，那么输入的用户名和密码无论是大写还是小写，只要正确就可以登录。应用了 strcmp()函数之后就避免了这种情况，即使正确，必须大小写匹配才可以登录，从而提高了网站的安全性。

4.8.2 按自然排序法比较

在 PHP 中，按照自然排序法进行字符串比较是通过 strnatcmp()函数来实现的。自然排序法比较的是字符串中的数字部分，将字符串中的数字按照大小进行比较。

strnatcmp()函数通过自然排序法比较字符串。

语法：

```
int strnatcmp ( string str1, string str2)
```

如果字符串相等，则返回 0；如果参数 str1 大于参数 str2，则返回 1；如果参数 str1 小于参数 str2，则返回-1。本函数区分字母大小写。

注意 在自然排序法中，2 比 10 小。在计算机序列当中，10 比 2 小，因为“10”中的第一个数字是“1”，它小于 2。

例 4.11 应用 strnatcmp()函数按自然排序法进行字符串的比较。（**实例位置：光盘\TM\Instance\04\4.11**）

代码如下：

```
<?php
$str1="str2.jpg";                                 //定义字符串常量
$str2="str10.jpg";                                //定义字符串常量
$str3="mrsoft1";                                  //定义字符串常量
$str4="MRSOFT2";                                  //定义字符串常量
echo strcmp($str1,$str2);                         //按字节进行比较，返回 1
echo strcmp($str3,$str4);                         //按字节进行比较，返回 1
echo strnatcmp($str1,$str2);                      //按自然排序法进行比较，返回-1
echo strnatcmp($str3,$str4);                      //按自然排序法进行比较，返回 1
?>
```

结果为：1 1 -1 1。

说明 按照自然排序法进行比较，还可以使用另一个与 strnatcmp()函数相同，但是不区分大小写的 strnatcasecmp()函数，它的作用和 strnatcmp()函数相同。

4.8.3　指定从源字符串的位置比较

strncmp()函数用来比较字符串中的前 n 个字符。

语法：

```
int strncmp(string str1,string str2,int len)
```

如果字符串相等，则返回 0；如果参数 str1 大于参数 str2，则返回 1；如果参数 str1 小于参数 str2，则返回-1。该函数区分字母大小写。参数说明如表 4.2 所示。

表 4.2　strncmp()函数的参数说明

参　数	说　明
str1	指定参与比较的第一个字符串对象
str2	指定参与比较的第二个字符串对象
len	必选参数，用来指定每个字符串中参与比较字符的数量

例 4.12　应用 strncmp()函数比较字符串的前 5 个字符是否与源字符串相等。**（实例位置：光盘\TM\Instance\04\4.12）**

代码如下：

```
<?php
$str1="I like life !";                          //定义字符串常量
$str2="i am a teacher!";                        //定义字符串常量
echo strncmp($str1,$str2,5);                    //比较前 6 个字符
?>
```

结果为：-1。

从上面的代码中可以看出，由于变量$str2 中的字符串的首字母为小写，与变量$str1 中的字符串不匹配，因此比较后的字符串返回值为-1。

4.9　检索字符串

在 PHP 中，提供了很多用于字符串查找的函数，PHP 也可以像 Word 那样实现对字符串的查找功能。下面讲解常用的字符串检索技术。

4.9.1　使用 strstr()函数检索指定的关键字

获取一个指定字符串在另一个字符串中首次出现的位置到后者末尾的子字符串。如果执行成功，则返回获取的子字符串（存在相匹配的字符）；如果没有找到相匹配的字符，则返回 false。

语法：

```
string strstr ( string haystack, string needle)
```

参数说明如表 4.3 所示。

表 4.3　strstr()函数的参数说明

参　数	说　明
haystack	必选参数，用来指定从哪个字符串中进行搜索
needle	必选参数，用来指定搜索的对象，如果该参数是一个数值，那么将搜索与这个数值的 ASCII 值相匹配的字符



本函数区分字母的大小写。

例 4.13　应用 strstr()函数获取指定字符串在字符串中首次出现的位置后的所有字符。（**实例位置：光盘\TM\Instance\04\4.13**）

代码如下：

```
<?php
echo strstr("明日科技图书网","图");                              //输出查询的字符串
echo "<br>";                                                     //执行换行
echo strstr("http://www.mingribook.com","m");                    //输出查询的字符串
echo "<br>";                                                     //执行换行
echo strstr("0431-84978981","8");                                //输出查询的字符串
?>
```

结果为：图书网

mingribook.com

84978981。

通过上面的代码可以看出，应用 strstr()函数自定义检索字符串非常方便，另外，strrchr()函数与其正好相反，该函数是从字符串后序的位置开始检索子串的。

4.9.2　应用 substr_count()函数检索子串出现的次数

在 PHP 中，有一种技术可以获取字符串中字符和单词数量，通过该技术可以查看到指定字符或者单词在字符串中出现的次数，而且还可以应用到论坛、博客或者聊天室的信息发布模块中，判断提交的信息中是否含有非法关键字。

substr_count()函数获取指定字符在字符串中出现的次数。

语法：

```
int substr_count(string haystack,string needle)
```

参数 haystack 是指定的字符串，参数 needle 为指定的字符。

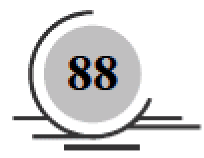
例 4.14　下面应用 substr_count()函数获取子串在字符串中出现的次数。（**实例位置：光盘\TM\Instance\04\4.14**）

代码如下：

```
<?php
$str="明日科技图书网图书";
echo substr_count($str,"书");                                    //输出查询的字符串
?>
```

结果为：2。

从表面上看，该函数的功能就是获取指定字符在字符串中出现的次数，输出的两个数字，但在实际的运用中，只要对输出的数字加以判断后，就能够实现不同功能。



技巧 检索子串出现的次数一般常用于搜索引擎中，针对子串在字符串中出现的次数进行统计，便于用户第一时间掌握子串在字符串中出现的次数。

4.10 替换字符串

通过字符串的替换技术可以实现对指定字符串中的指定字符进行替换。字符串的替换技术可以通过以下两个函数实现：str_ireplace()和 substr_replace()函数。

1. str_irereplace()函数

使用新的子字符串替换原始字符串中被指定要替换的字符串。

语法：

```
mixed str_ireplace ( mixed search, mixed replace, mixed subject [, int &count] )
```

将所有在参数 subject 中出现的 search 用参数 replace 替换，参数&count 表示替换字符串执行的次数。str_replace()函数的参数说明如表 4.4 所示。

表 4.4 str_replace()函数的参数说明

参 数	说 明
search	必选参数，指定需要查找的字符串
replace	必选参数，指定替换的值
subject	必选参数，指定查找的范围
count	可选参数，获取执行替换的数量

注意 该函数可以以数组的方式传递参数。函数返回的是一个字符串还是数组，取决于被操作的对象是字符串还是数组。如果原始字符串 subject 是一个数组，则该函数会依次用 replace 替换 subject 数组中每个元素中的 search 子字符串，同时该函数的返回值为一个数组。

例 4.15 将文本中的指定字符串“你好”替换为“明日科技”，并且输出替换后的结果。（**实例位置：光盘\TM\Instance\04\4.15**）

实例代码如下：

```
<?php
$str1 = "你好";                                   //定义字符串常量
$str2 = "明日科技";                               //定义字符串常量
$str = “你好公司是一家以编程词典技术为核心的高科技企业，多年来始终致力于图书软件的开发、编程词典的销
售、网站的访问日益增多";                          //定义字符串常量
Echo str_ireplace($str1,$str2,$str);              //输出替换后的字符串
?>
```

运行结果如图 4.6 所示。

注意 该函数在执行替换的操作时不区分大小写，如果需要对大小写加以区分，可以使用 str_replace()函数。

字符串替换技术最常用的就是在搜索引擎的关键字处理中，可以使用字符串替换技术将搜索到的字符串中的关键字替换颜色，如查询关键字描红功能，使搜索到的结果更便于用户查看。

注意 查询关键字描红是指将查询关键字以特殊的颜色、字号或字体进行标识。这样可以使浏览者快速检索到所需的关键字，方便浏览者从搜索结果中查找所需内容。查询关键字描红适用于模糊查询。

例 4.16 使用 str_ireplace()函数替换查询关键字，当显示所查询的相关信息时，将输出的关键字的字体替换为红色。**（实例位置：光盘\TM\Instance\04\4.16）**

实例代码如下：

```
<?php
    $c = "吉林省明日科技有限公司，是一家知名的软件公司，明日主要推出编程词典，及图书等";
    $str="明日科技";
    echo str_ireplace($str,"<font color='#ff0000'>".$str."</font>",$c);
?>
```

运行结果如图 4.7 所示。



图 4.6 应用 str_replace()函数替换子字符串

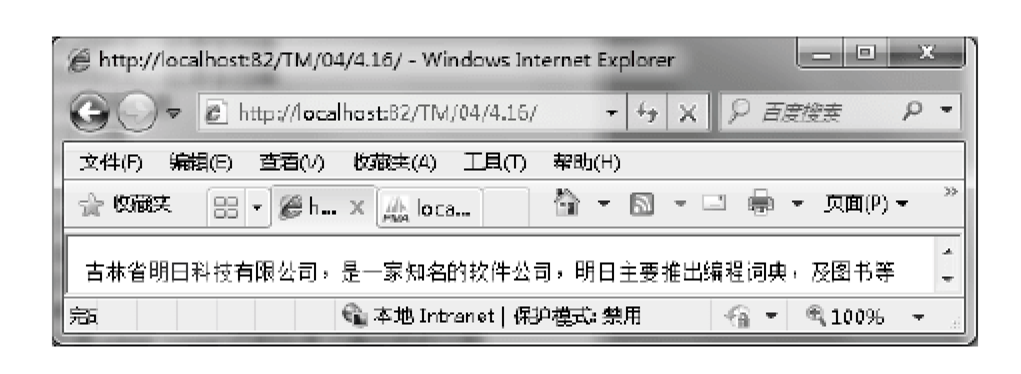


图 4.7 应用 str_ireplace()函数对查询关键字描红

2. substr_replace()函数

对指定字符串中的部分字符串进行替换。

语法：

```
mixed substr_replace ( string str,string repl, int start,[int length])
```

参数说明如表 4.5 所示。

表 4.5 substr_replace()函数的参数说明

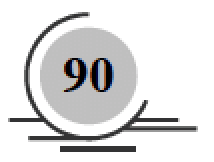
参　　数	说　　明
str	指定要操作的原始字符串
repl	指定替换后的新字符串
start	指定替换字符串开始的位置。正数表示起始位置从字符串开头开始；负数表示起始位置从字符串的结尾开始；0 表示起始位置字符串中的第一个字符
length	可选参数，指定替换的字符串长度。默认值是整个字符串。正数表示起始位置从字符串开头开始；负数表示起始位置从字符串的结尾开始；0 表示插入而非替代

注意 如果参数 start 设置为负数，而参数 length 数值小于或等于 start 数值，那么 length 的值自动为 0。

例 4.17 下面将指定字符串中的“双倍”替换为“百倍”，并且输出替换后的结果。**（实例位置：光盘\TM\Instance\04\4.17）**

实例代码如下：

```
<?php
$str="用今日的辛勤工作，换明日的双倍回报！";                    //定义字符串常量
```



```
$replace="百倍";                                    //定义要替换的字符串
echo substr_replace($str,$replace,26.4);            //替换字符串
?>
```

结果为：用今日的辛勤工作，换明日的百倍回报！

4.11　什么是正则表达式

正则表达式是一种描述字符串结构的语法规则，是一个特定的格式化模式。它可以匹配、替换、截取匹配的字串。对于用户来说，可能以前接触过 DOS，如果想匹配当前文件夹下所有的文本文件，可以输入“dir *.txt”命令，按 Enter 键后，所有“.txt”文件将会被列出来。这里的“*.txt”就可以理解为一个简单的正则表达式。

在学习正则表达式前，我们先来了解一个正则表达式中的几个容易混淆的术语，这对于学习正则表达式有很大的帮助。

☑ grep：最初是 ED 编辑器中的一条命令，用来显示文件中特定的内容。后来成为一个独立的工具 grep。

☑ egrep：grep 虽然不断地更新升级，但仍然无法跟上技术的脚步。为此，贝尔实验室写出了 egrep，意为“扩展的 grep”。这大大增强了正则表达式的能力。

☑ POSIX（Portable Operating System Interface of UNIX）：可移植操作系统接口。在 grep 发展的同时，其他一些开发人员也根据自己的喜好开发出了具有独特风格的版本。但问题也随之而来。有的程序支持某个元字符，而有的程序则不支持。因此，就有了 POSIX。POSIX 是一系列标准，确保了操作系统之间的移植性。不过 POSIX 和 SQL 一样，没有成为最终的标准而只能作为一个参考。

☑ Perl（Practical Extraction and Reporting Language）：实际抽取与汇报语言。1987 年，Larry Wall 发布了 Perl。在随后的 7 年时间里，从 Perl1 到现在的 Perl5，最终成为了 POSIX 之后的另一个标准。

☑ PCRE：Perl 的成功，让其他的开发人员在某种程度上要兼容“Perl”，包括 C/C++、Java、Python 等都有自己的正则表达式。1997 年，Philip Hazel 开发了 PCRE 库，这是兼容 Perl 正则表达式的一套正则引擎，其他开发人员可以将 PCRE 整合到自己的语言中，为用户提供丰富的正则功能。许多软件都使用 PCRE，PHP 正是其中之一。

4.12　正则表达式语法规则

视频讲解：光盘\TM\Video\第 4 章\视频 4.12\正则表达式语法规则.exe

正则表达式由两部分构成：元字符和文本字符。元字符就是具有特殊含义的字符，如前面提到的“*”和“？”。文本字符就是普通的文本，如字母和数字等。PCRE 风格的正则表达式一般都放置在定界符“/”中间，如“/\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*/”、“/^http:\/\/(www\.)?.+.?$/”。为了便于读者理解，除个别实例外，本节中的表达式不给出定界符“/”。

4.12.1　行定位符（^和$）

行定位符用于描述字串的边界。“^”表示行的开始；“$”表示行的结尾。例如：

```
^mingri
```

该表达式表示要匹配字串 mr 的开始位置是行头，如“mrsoft”、“mrbook”都可以匹配，而“Tomorrow mr”则不能匹配。如果要匹配“Tomorrow mr”，则可以使用下面的行结尾定位符“$”。

```
mingri$
```

这可以匹配以 mingri 结尾的字符，而不能匹配以 mingri 开头的字符，如果要匹配的字串可以出现在字符串的任意部分，那么可以直接写成：

```
mingri
```

这样两个字符串就都可以匹配了。

4.12.2 字符类（[ ]）

正则表达式是区分大小写的，如果想要忽略大小写，可以通过方括号（[]）表达式来完成。只要匹配的字符出现在方括号内，即表示匹配成功。但需要注意的是，一个方括号只能匹配一个字符。例如，要匹配字串“mingri”不区分大小写，其表达式的格式如下。

```
[MINGming][Riri]
```

这样，就可以匹配字串“mingri”的所有写法。POSIX 和 PCRE 都使用了一些预定义字符类，但表示方法略有不同。POSIX 风格的预定义字符类如表 4.6 所示。

表 4.6 POSIX 预定义字符类

预定义字符类	说　明
[:digit:]	十进制数字集合。等同于[0-9]
[[:alnum:]]	字母和数字的集合。等同于[a-zA-Z0-9]
[[:alpha:]]	字母集合。等同于[a-zA-Z]
[[:blank:]]	空格和制表符
[[:xdigit:]]	十六进制数字
[[:punct:]]	特殊字符集合。包括键盘上的所有特殊字符，如!@#$?等
[[:print:]]	所有的可打印字符（包括空白字符）
[[:space:]]	空白字符（空格、换行符、换页符、回车符、水平制表符）
[[:graph:]]	所有的可打印字符（不包括空白字符）
[[:upper:]]	所有大写字母。[A-Z]
[[:lower:]]	所有小写字母。[a-z]
[[:cntrl:]]	控制字符

而 PCRE 的预定义字符类则是使用反斜杠来表示的。

4.12.3 选择字符（|）

要忽略字串的大小写，还可以通过选择字符（|）来完成。该字符可以理解为“或”，如上例也可以写成：

```
M|mR|r
```

该表达式的意思是以字母 M 或 m 开头，后面接一个字母 R 或 r。


说明 使用“[]”和使用“|”的区别在于：“[]”只能匹配单个字符，而“|”可以匹配任意长度的字串。

4.12.4 连字符（-）

变量的命名规则是只能以字母和下划线开头。如果要使用正则表达式来匹配变量名的第一个字母，要

写成如下格式。

```
[a,b,c,d…A,B,C,D…_]
```

这实在是非常麻烦，但正则表达式提供了连字符“-”来解决这个问题。连字符可以表示字符的范围。如上例可以写成：

```
[a-zA-Z_]
```

4.12.5　排除字符（[^]）

上面的例子是匹配符合命名规则的变量。反过来，匹配不符合命名规则的变量。正则表达式提供了“^”字符。这个元字符在 4.12.1 节中曾经出现过，表示行的开始。这里将其放到方括号中，则表示排除的意思。例如：

```
[^a-zA-Z_]
```

该表达式匹配的是不以字母和下划线开头的变量名。

4.12.6　限定符（? * + {n,m}）

经常使用 google 的用户可能会发现，在搜索结果页的下方，“google”中间字母“o”的个数会随着搜索页的改变而变化。那么要匹配该字串的正则表达式该如何实现呢？

对于这类重复出现的字母或字串，可以使用限定符来实现匹配。限定符主要有 6 种，如表 4.7 所示。

表 4.7　限定符的说明和举例

限　定　符	说　　明	举　　例
?	匹配前面的字符零次或一次	colou?r，该表达式可以匹配 colour 和 color
+	匹配前面的字符一次或多次	go+gle，该表达式可以匹配的范围从 gogle 到 goo…gle
*	匹配前面的字符零次或多次	go*gle，该表达式可以匹配的范围从 ggle 到 goo…gle
{n}	匹配前面的字符 n 次	go{2}gle，该表达式只匹配 google
{n,}	匹配前面的字符最少 n 次	go{2,}gle，该表达式可以匹配的范围从 google 到 goo…gle
{n,m}	匹配前面的字符最少 n 次，最多 m 次	employe{0,2}，该表达式可以匹配 employ、employe 和 employee 3 种情况

通过对 google 应用的观察发现，当搜索结果只有一页时，不显示 google 标志；只有大于等于 2 时，才显示 google。这说明字母“o”最少为两个，而最多可以看到 20 个，那么其正则表达式为：

```
go{2,20}gle
```

4.12.7　点字符（.）

在正则表达式中通过点字符“.”可以匹配除换行符外的任意一个字符。注意，是除换行符外的任意的一个字符。

例 4.18　在本例中应用正则表达中的点字符（.）匹配变量中是否包含指定的字符串。**（实例位置：光盘\TM\Instance\04\4.18）**

程序代码如下：

```
<?php
	$str="吉林省明日科技有限公司";
	if(preg_match("/^明日.科技$/",$str,$counts)){				//判断变量中是否包含明日、科技字符串
		echo "<script>alert('没有指定的字符串!'); history.back();</script>";
```

```
    }else{
        echo "<script>alert('存在指定的字符串!'); history.back();</script>";
    }
?>
```

结果为：存在指定的关键字!

4.12.8 反斜杠（\）

除了可以做转义字符外，反斜杠还有其他一些功能。

☑ 反斜杠可以将一些不可打印的字符显示出来，如表 4.8 所示。

表 4.8 反斜杠显示的不可打印字符

字 符	说 明
\a	警报，即 ASCII 中的<BEL>字符（0x07）
\b	退格，即 ASCII 中的<BS>字符（0x08）。注意，在 PHP 中只有在中括号（[ ]）里使用才表示退格
\e	escape，即 ASCII 中的<ESC>字符（0x1B）
\f	换页符，即 ASCII 中的<FF>字符（0x0C）
\n	换行符，即 ASCII 中的<LF>字符（0x0A）
\r	回车符，即 ASCII 中的<CR>字符（0x0D）
\t	水平制表符，即 ASCII 中的<HT>字符（0x09）
\xhh	十六进制代码
\ddd	八进制代码
\cx	即 control-x 的缩写，匹配由 x 指明的控制字符，其中 x 是任意字符

☑ 还可以指定预定义字符集，如表 4.9 所示。

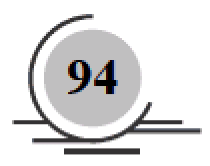
表 4.9 反斜杠指定的预定义字符集

限 定 符	说 明
\d	任意一个十进制数字，相当于[0-9]
\D	任意一个非十进制数字
\s	任意一个空白字符（空格、换行符、换页符、回车符、水平制表符），相当于[\f\n\r\t]
\S	任意一个非空白字符
\w	任意一个单词字符，相当于[a-zA-Z0-9_]
\W	任意一个非单词字符

☑ 反斜杠还有一种功能，就是定义断言，其中已经接触过了\b、\B，其他如表 4.10 所示。

表 4.10 反斜杠定义断言的限定符

限 定 符	说 明
\b	单词分界符，用来匹配字符串中的某些位置，\b 是以统一的分界符来匹配
\B	非单词分界符序列
\A	总是能够匹配待搜索文本的起始位置
\Z	表示在未指定任何模式下匹配的字符，通常是字符串的末尾位置，或者是在字符串末尾的换行符之前的位置
\z	只匹配字符串的末尾，而不考虑任何换行符
\G	当前匹配的起始位置



4.12.9　反向引用

反向引用，就是依靠子表达式的“记忆”功能来匹配连续出现的字串或字母。例如，匹配连续两个“it”，首先将单词“it”作为分组，然后在后面加上“\1”即可。格式为：

```
(it)\1
```

这就是反向引用最简单的格式。如果要匹配的字串不固定，那么就将括号内的字串写成一个正则表达式。如果使用了多个分组，那么可以用“\1”、“\2”来表示每个分组（顺序是从左到右）。例如：

```
([a-z])([A-Z])\1\2
```

除了可以使用数字来表示分组外，还可以自己指定分组名称。格式为：

```
(?P<subname>…)
```

如果想要反向引用该分组，使用如下语法。

```
(?P=subname)
```

下面重写一下表达式([a-z])([A-Z])\1\2，为这两个分组分别命名，并反向引用它们。正则表达式如下。

```
 (?P<fir>[a-z])(?P<sec>[A-Z])(?P=fir)(?P=sec)
```

4.13　POSIX 扩展正则表达式函数

视频讲解：光盘\TM\Video\第 4 章\视频 4.13\POSIX 扩展正则表达式函数.exe

PHP 中实现 POSIX 正则表达式的函数有 7 个。下面就来了解其中几个常用函数。

4.13.1　替换字符串

函数格式：

```
string ereg_replace/eregi_replace ( string pattern, string replacement, string string )
```

函数功能：在字符串 string 中匹配表达式 pattern。如果匹配成功，则使用 replacement 来替换匹配字串，并返回替换后的 string。如果未在 string 中找到匹配项，则 string 将原样返回。eregi_replace()函数不区分大小写。

例 4.19　将字符串中所有非大写的“mingri”都换成大写“MINGRI”。（**实例位置：光盘\TM\Instance\04\4.19**）

实例代码如下：

```
<?php
     $ereg = 'mingri';                                    //要匹配的字串
     $str = 'nihao,MINGri,mingri,mingRI.';                //要查找的文本
     $rep_str = eregi_replace($ereg,'MINGRI',$str);       //使用 eregi_replace()函数进行替换
     echo $rep_str;                                       //输出替换后的文本
?>
```

结果为：nihao,MINGRI,MINGRI,MINGRI.

4.13.2　分割字符串

函数格式：

```
array split/spliti ( string pattern, string string [, int limit] )
```

函数功能：使用表达式 pattern 来分割字符串 string。如果有参数 limit，那么数组最多有 limit 个元素，剩余部分都写到最后一个数组元素中。如果函数错误，则返回 false。split()函数区分大小写，而 spliti()函数

不区分大小写。

例 4.20 通过 spliti()函数，使用字串“、”来分割字符串$str。（**实例位置：光盘\TM\Instance\04\4.20**）

代码如下：

```
<?php
      $ereg = '、';                                                   //分割字符串使用的表达式
      $str = 'JAVA 编程词典、VB 编程词典、VC 编程词典、PHP 编程词典';    //要被分割的字符串
      $arr_str = spliti($ereg,$str);                                  //使用 spliti()函数分割字符串
      var_dump($arr_str);                                             //显示分割后的数组结构
?>
```

结果为：array(4) { [0]=> string(15) "PHP 编程词典" [1]=> string(13) "C 编程词典" [2]=> string(15) "ASP 编程词典" [3]=> string(16) "JAVA 编程词典" }。

4.14 PCRE 兼容正则表达式函数

视频讲解：光盘\TM\Video\第 4 章\PCRE 兼容正则表达式函数.exe

实现 PCRE 风格的正则表达式的函数也有 7 个。但无论从执行效率，还是从语法支持上，PCRE 函数都要略优于 POSIX 函数。下面对其中几个常用的 PCRE 函数进行讲解。

4.14.1 查找字符串

函数格式：

int preg_match/preg_match_all (**string pattern**, **string subject [, array matches]**)

函数功能：在字符串 subject 中匹配表达式 pattern。函数返回匹配的次数。如果有数组 matches，那么每次匹配的结果将被存储到数组 matches 中。

函数 preg_match()的返回值是 0 或 1。因为该函数在匹配成功后就停止继续查找了。而 preg_match_all()函数则会一直匹配到最后才停止。参数 array matches 对于 preg_match_all()函数是必需的，而对前者则可以省略。

例 4.21 使用 preg_match()和 preg_match_all()函数验证手机和座机号码的格式是否正确，并返回各自的匹配次数。（**实例位置：光盘\TM\Instance\04\4.21**）

实例代码如下：

```
<?php
      $str='this is a mingribook luntan xiaoxi';
      $preg='/\b\w{6}\b/';
      $num1=preg_match($preg,$str,$str1);
      echo $num1.'<br>';
      var_dump($str1);
      $num2 =preg_match_all($preg,$str,$str2);
      echo '<p>'.$num2.'<br>';
      var_dump($str2);
?>
```

运行结果如图 4.8 所示。

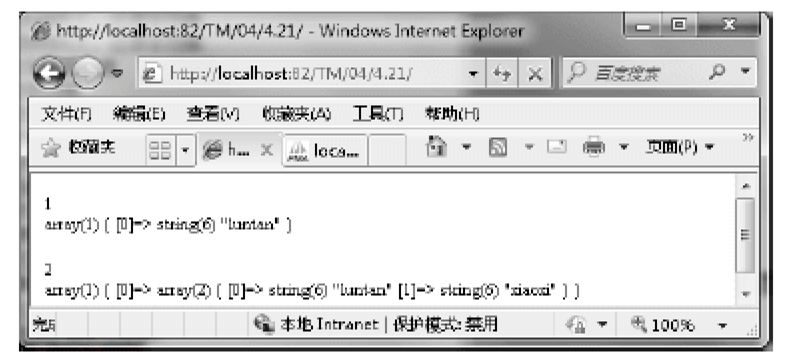
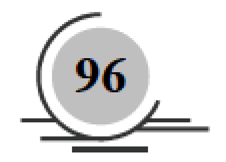


图 4.8 preg_match()和 preg_match_all()函数

4.14.2 替换字符串

函数语法：



```
mixed preg_replace ( mixed pattern, mixed replacement, mixed subject [, int limit] )
```

函数功能：该函数在字符串 subject 中匹配表达式 pattern，并将匹配项替换成字串 replacement。如果有参数 limit，则替换 limit 次。

例 4.22　将输入的“[b]…[/b]”、“[i]..[/i]”等类似的格式转换为 html 能识别的标签。**（实例位置：光盘\TM\Instance\04\4.22）**

实例代码如下：

```
<?php
	$string = '[b]粗体字[/b]';
	$b_rst = preg_replace('/\[b\](.*)\[\/b\]/i','<b>$1</b>',$string);
	echo $b_rst;
?>
```

结果为：粗体字。

> **说明**
> preg_replace()函数中的字串“$1”是在正则表达式外调用分组，按照$1、$2 排列，依次表示从左到右的分组顺序，也就是括号顺序。另外，$0 表示的是整个正则表达式的匹配值。

4.15　实　　战

视频讲解：光盘\TM\Video\第 4 章\实战.exe

4.15.1　超长文本的分页显示

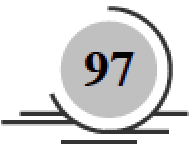
例 4.23　本实例实现超长文本的分页输出，其中通过 strlen()函数获取字符串的长度，应用 ceil()函数将文本文件中的数据分页；通过自定义函数 msubstr()控制数据的输出；通过 substr()函数获取到当前页面输出的内容。**（实例位置：光盘\TM\Instance\04\4.23）**

在程序开发的过程中，经常会遇到输出超长文本文件的情况，为了使页面的布局更加合理，文本文件的输出更加流畅，采用分页输出文本文件是一个很好的方法。运行本实例，如图 4.9 所示，通过分页来输出超长文本文件。



图 4.9　新闻主题信息显示

（1）编写 function.php 文件，创建自定义函数 unhtml()对超长文本中的特殊字符进行替换；创建自定义



函数 msubstr()控制超长文本中数据的输出。语法如下：

```
<?php
    function unhtml($content){                                    //定义自定义函数的名称
        $content=htmlspecialchars($content);                      //转换文本中的特殊字符
        $content=str_replace(chr(13),"<br>",$content);            //替换文本中的换行符
        $content=str_replace(chr(32)," ",$content);          //替换文本中的 
        $content=str_replace("[_[","<",$content];                 //替换文本中的大于号
        $content=str_replace(")_)",">",$content);                 //替换文本中的小于号
        $content=str_replace("|_|"," ",$content);                 //替换文本中的空格
        return trim($content);                                    //删除文本中首尾的空格
    }
    function msubstr($str,$start,$len){ //$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的是长度
        $strlen=$start+$len;         //用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
        for($i=0;$i<$strlen;$i++){              //通过 for 循环语句，循环读取字符串
            if(ord(substr($str,$i,1))>0xa0){    //如果字符串中首个字节的 ASCII 序数值大于 0xa0，则表示为汉字
                $tmpstr.=substr($str,$i,2); //每次取出两位字符赋给变量$tmpstr，等于一个汉字
                $i++;                       //变量自加 1
            }else{                          //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
                $tmpstr.=substr($str,$i,1);
            }
        }
        return $tmpstr;                     //输出字符串
    }
?>
```

（2）创建 index.php 页面，首先定义分页变量 page，通过 include 包含语句调用 function.php 文件。然后通过 file_get_contents()函数读取根目录下 file 文件夹中 file.txt 文件中的数据，实现数据的分页输出。其关键代码如下：

```
<?php if ($page=="") {$page=1;};    include("function.php");?>
<?php
//读取超长文本中的数据，实现超长文本中数据的分页显示
if($page){
    $counter=file_get_contents("file/file.txt");              //读取指定文件夹下的文件
    $length=strlen(unhtml($counter));                         //获取数据的长度，应用 unhtml()函数去除特殊字符
    $page_count=ceil($length/950);                            //计算共有多少页
    $c=msubstr($counter,0,($page-1)*950);                     //获取上一页的数据
    $c1=msubstr($counter,0,$page*950);                        //获取当前页的数据
    echo substr($c1,strlen($c),strlen($c1)-strlen($c));       //输出当前页的数据
  }
?>
```

（3）设置分页使用的超链接，定义链接的标识。其代码如下：

```
<tr>
    <td  width="202"  height="22"><span  class="STYLE3"> 页 次 ： <?php  echo  $page;?>  /  <?php  echo
$page_count;?>  页  </span></td>
     <td width="278"><span class="STYLE3">分页：
     <?php
        if($page!=1){
            echo   "<a href=index.php?page=1>首页</a> ";
            echo "<a href=index.php?page=".($page-1).">上一页</a> ";
        }
        if($page<$page_count){
            echo "<a href=index.php?page=".($page+1).">下一页</a> ";
            echo "<a href=index.php?page=".$page_count.">尾页</a>";
        }
```

```
        ?>
        </span></td>
</tr>
```

4.15.2　控制页面中输出字符串的长度

例 4.24　本实例应用 strlen()函数获取字符串的长度，在输出字符串时进行判断，如果字符串超出指定的长度，则使用指定的字符进行替换，并在输出时间字符串时，应用 str_replace()函数将“-”替换为“/”。**（实例位置：光盘\TM\Instance\04\4.24）**

在论坛或者电子商务等网站中，经常会输出一些公告信息、最新动态等内容，这些内容都是以标题的形式进行输出，为标题设置超链接，链接到相关内容的详细信息页面。

在输出标题信息时，由于要考虑页面规范化、设计合理，所以要对输出的标题长度进行限制，如果标题的总长度超出指定范围，就需要使用省略号进行替换。运行结果如图 4.10 所示。



图 4.10　控制页面中输出字符串的长度

（1）创建 index.php 页面，首先通过 include_once 语句调用数据库连接文件和字符串处理文件，然后执行查询语句，查询出数据表中的记录，最后输出最新动态的标题和发布时间，并判断如果标题的内容超过 24 个字节，则输出省略号，截取发布时间。其关键代码如下：

```
<?php
    include_once("conn/conn.php"); //调用连接数据库的文件
    include_once("function.php");
?>
<table width="365" height="22" border="0" align="center" cellpadding="0" cellspacing="0">
<?php
    $sql=mysql_query("select * from student order by age desc ",$id);
    while($myrow=mysql_fetch_array($sql)){
?>
    <tr>
        <td width="20" height="22"><div align="center"><img src="images/01.JPG"/></div></td>
        <td width="258" height="22"><a href="new_dynamic.php?id=<?php echo $myrow["id"];?>">
        <?php
            echo unhtml(msubstr($myrow["name"],0,24));
            echo '   ';
            echo unhtml(msubstr($myrow["zhuanye"],0,24));
```

```
            ?>
            </a> </td>
            <td width="87">
                <?php
                    echo"<fontcolor=red>（".substr(str_replace("-","/",$myrow[address]),0,10)."）</font>";
                ?>
            </td>
        </tr>
        <tr><td colspan="3"></td></tr>
        <?php   }   ?>
</table>
```

（2）创建 new_dynamic.php 页面，根据超链接中传递的 ID 值，从数据表中查询出指定的记录，并输出记录的详细内容。其关键代码如下：

```
<?php
    include_once("conn/conn.php"); //调用连接数据库的文件
    include_once("function.php");
?>
<?php
    $sql=mysql_query("select * from student where id='".$_GET[id]."'",$id);
    while($myrow=mysql_fetch_array($sql)){
?>
<td align="left" valign="top" class="STYLE1"><span class="STYLE1">
    <?php echo $myrow[name];?> <br><br>
    <?php echo $myrow[zhuanye];?></span></td>
<?php }?>
```

4.15.3　正则无刷新用户注册

例 4.25　通过正则表达式无刷新验证用户注册信息是否合理。正则表达式经常被应用于对用户注册信息的合理性判断中，可以对用户输入的邮编、电话号码、邮箱地址和网址的格式进行判断。本实例中应用正则表达式和 JavaScript 脚本，实现无刷新判断用户输入信息的格式是否正确。**（实例位置：光盘\TM\Instance\04\4.25）**

（1）本例中将正则表达式定义到自定义函数中，然后在 JavaScript 脚本中调用自定义函数，执行对用户输入信息的判断。JavaScript 脚本文件 check.js 的代码如下：

```
function checkregtel(regtel){                          //通过正则表达式验证手机号码的格式是否正确
    var str=regtel;
    var Expression=/^13(\d{9})$|^15(\d{9})$/;          //定义正则表达式
    var objExp=new RegExp(Expression);
    if(objExp.test(str)==true){
        return true;
    }else{
        return false;
    }
}
function checkregtels(regtels){                        //自定义函数，验证座机号码的格式是否正确
    var str=regtels;
    var Expression=/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$/;    //定义正则表达式
    var objExp=new RegExp(Expression);
    if(objExp.test(str)==true){
        return true;
    }else{
        return false;
    }
```

```
}
function checkregemail(emails){                    //自定义函数，验证 E-mail 地址的格式是否正确
    var str=emails;
    var Expression=/\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*/;          //定义正则表达式
    var objExp=new RegExp(Expression);
    if(objExp.test(str)==true){
        return true;
    }else{
        return false;
    }
}
function chkreginfo(form,mark){                    //自定义函数，对表单中用户提交的数据进行判断
//这里省略了部分代码
    if(mark==4 || mark=="all"){                    //验证 E-mail 地址的格式是否正确
        if(form.email.value==""){                  //如果输入值为空
            chknew_email.innerHTML="请输入 E-mail 地址！";          //则返回“请输入 E-mail 地址！”
            form.email.style.backgroundColor="#FF0000";           //定义返回提示信息的样式
            return false;
        }else if(!checkregemail(form.email.value)){       //调用自定义函数验证 E-mail 地址格式
            chknew_email.innerHTML="邮箱地址的格式不正确！";
            form.email.style.backgroundColor="#FF0000";
            return false;
        }else{
            chknew_email.innerHTML="";                        //当返回值是空时
            form.email.style.backgroundColor="#FFFFFF";       //定义样式为白色
        }
    }
    if(mark==5 || mark=="all"){                                //验证手机号码的格式
        if(form.mtel.value==""){
            chknew_mtel.innerHTML="请输入电话号码！";
            form.mtel.style.backgroundColor="#FF0000";
            return false;
        }else if(!checkregtel(form.mtel.value)){              //调用自定义函数验证手机号码格式是否正确
            chknew_mtel.innerHTML="电话号码的格式不正确！";
            form.mtel.style.backgroundColor="#FF0000";
            return false;
        }else if(isNaN(form.mtel.value)){
            chknew_mtel.innerHTML="电话号由数字组成！";
            form.mtel.style.backgroundColor="#FF0000";
            return false;
        }else{
            chknew_mtel.innerHTML="";
            form.mtel.style.backgroundColor="#FFFFFF";
        }
    }
    //省略了部分代码
}
```

（2）正则表达式和 JavaScript 脚本文件创建完成后，接下来开始创建 index.php 文件，添加 form 表单，调用 JavaScript 脚本文件，提交用户注册信息。其关键代码如下：

```
<link rel="stylesheet" type="text/css" href="css/style.css">
<script src="js/check.js"></script>
<form name="form_reg" method="post" action="index_ok.php" onSubmit="return chkreginfo(form_reg,'all')">
<table width="620" height="262" border="0" align="center" cellpadding="0" cellspacing="0">
```

```
        <tr><td width="120" height="30"><div align="right">用户名：</div></td>
            <td colspan="3"> 
                <input type="text" name="recuser" size="20" class="inputcss" onBlur="chkreginfo(form_reg,0)">
                <div id="chknew_recuser" style="color:#FF0000"></div>
            </td>
        </tr>
        <!--省略了部分代码-->
        <tr><td height="30"><div align="right">E-mail：</div></td>
            <td height="30" colspan="3"> 
                <input type="text" name="email" size="20" class="inputcss" onBlur="chkreginfo(form_reg,4)">
                <div id="chknew_email" style="color:#FF0000"></div>
            </td>
        </tr>
        <!--省略了部分代码-->
        <tr><td height="30"> </td>
            <td height="30" colspan="2">
                <input type="image"   src="images/form (2).jpg">
                <img src="images/form.jpg" onClick="form_reg.reset()" style="cursor:hand"/>
            </td>
        </tr>
</table>
</form>
```

（3）创建 index_ok.php 文件，输出用户提交的数据。运行结果如图 4.11 所示。



图 4.11　通过正则表达式无刷新验证用户注册信息

4.15.4　计算密码长度

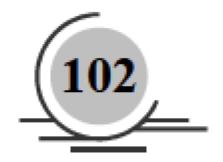
对密码的长度进行计算，以及对密码长度进行限制，在实际应用中比较广泛，尤其是实现会员注册功能。

例 4.26　本例介绍一种应用 strlen()函数计算输入的密码长度的方法，运行结果如图 4.12 所示。**（实例位置：光盘\TM\Instance\04\4.26）**

其实现的过程如下所示。

（1）通过 Dreamweaver 设计一个 form 表单，表单元素为一个提交密码的文本域，一个“计算”按钮。其代码如下：

```
<form action="" method="post" name="form1" class="STYLE1" id="form1">
<td width="411" bgcolor="#99CC33"><div align="center"><span class="STYLE2">密码：
```



```
            <input name="pass" type="password" size="12" />
        </span>
              <input type="submit" name="Submit" value="计算" />
</form>
```

（2）在本页中通过 isset()函数判断用户提交的密码和按钮值是否存在。如果存在则通过$_POST[]全局数组获取表单中提交的密码，通过 strlen()函数计算出密码的长度，并在页面上显示。其主要代码如下：

```
<?php
$pass="";
if(isset($_POST['pass'])&&$_POST['pass']!=""){              //对 form 表单的文本域进行判断
$pass=$_POST['pass'];
     echo strlen($pass);                                     //通过 strlen 函数计算文本域中的密码长度
}
?>
```

4.15.5　去除用户填写注册信息中的空格

用户在输入数据时，经常会在无意中输入多余的空白字符，在某些情况下，字符串中不允许出现空白字符和特殊字符，这就需要去除字符串中的空白字符和特殊字符。在 PHP 中提供 trim()函数去除字符串左右两边的空白字符和特殊字符，ltrim()函数去除字符串左边的空白字符和特殊字符，rtrim()函数去除字符串中右边的空白字符和特殊字符。

例 4.27　在本例中应用 ltrim()函数去除用户名和空格。效果如图 4.13 所示。**（实例位置：光盘\TM\Instance\04\4.27）**

```
<?php
$strs="";                                                    //声明关于文本域的变量
$pass="";
$tel="";
$add="";
if(isset($_POST['Submit']) && $_POST['Submit']=="提交"){     //对表单的提交按钮进行判断
     if(isset($_POST['user']) && $_POST['user']!=""){        //对用户名进行判断，非空时执行下面的操作
          $pass=$_POST['pass'];
          $tel=$_POST['tel'];
          $add=$_POST['add'];
          $str=$_POST['user'];                               //调用用户名文本框提交的值
          $strs=ltrim($str)."\n";                            //去除用户名左边的空格
     }else{
          echo "<script>alert('提交内容不能为空');</script>";
     }
}
?>
```

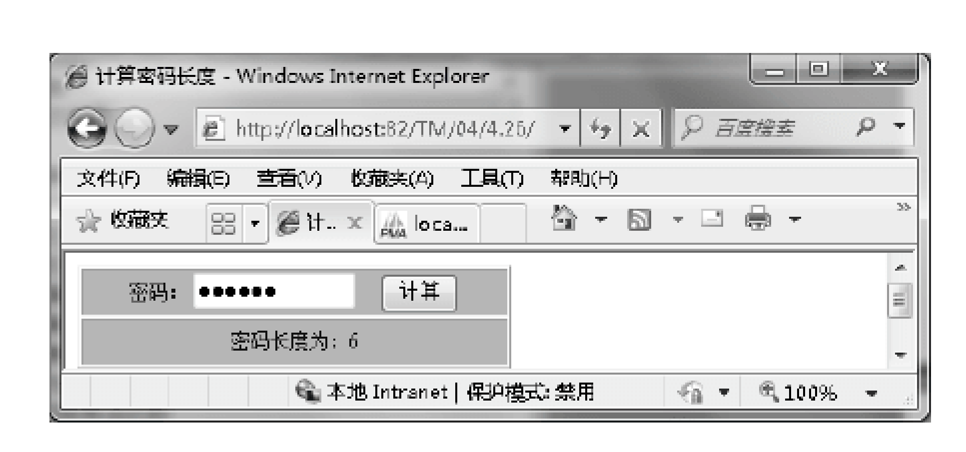


图 4.12　计算密码长度显示界面

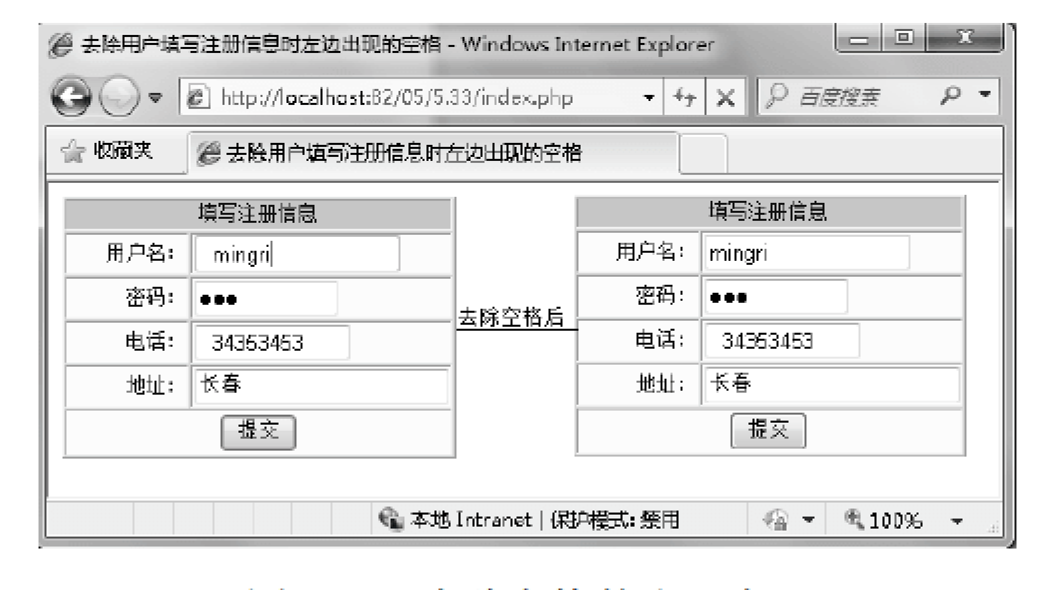


图 4.13　去除空格的注册表

4.16 小　　结

本章主要讲解了字符串的操作技术及正则表达式的用法，包括通过单引号、双引号和定界符标识字符串，字符串的连接、转义和还原，字符串的截取、比较、检索、替换和分割等，以及正则表达式的常用语法规则，并且介绍了正则表达式的两种主要风格；另外，通过实例、实战和实战练习对字符串和正则表达式的实际应用进行了讲解。希望通过本章的学习，读者可以对字符串操作和正则表达式有个初步的认识。

4.17 学习成果检验

1．尝试开发一个页面，验证用户输入的 15 位或 18 位身份证号长度是否正确。**（答案位置：光盘\TM\Instance\04\4.28）**

2．通过正则表达式验证邮箱地址的格式是否正确。**（答案位置：光盘\TM\Instance\04\4.29）**

3．使用正则表达式匹配 html 标签。**（答案位置：光盘\TM\Instance\04\4.30）**

第 5 章

初探数组

（视频讲解：146 分钟）

数组是对大量数据进行有效组织和管理的手段之一，通过数组的强大功能，可以对大量数据类型相同的数据进行存储、排序、插入及删除等操作，从而有效地提高程序开发效率及改善程序的编写方式。PHP 是市面上最为流行的 Web 开发语言之一，具有代码开源、升级速度快等特点，对数组的操作能力更为强大，尤其为程序开发人员提供了大量方便、易懂的数组操作函数，使 PHP 深受广大 Web 开发人员的青睐。

通过阅读本章内容，你可以：

- 熟悉数组的类型
- 了解如何输出数组
- 掌握如何遍历数组
- 掌握 PHP 的全局数组
- 掌握如何统计数组元素个数

5.1 什么是数组

数组是一组数据的集合，把一系列数组组织起来，形成一个可操作的整体。PHP 中的数组较为复杂，但比其他许多高级语言中的数组更灵活。数组 array 是一组有序的变量，其中每个变量称为一个元素。每个元素由一个特殊的标识符来区分，该标识符称为键(也称为下标)，变量中保存单个数据，数组中保存多个变量，如图 5.1 所示。

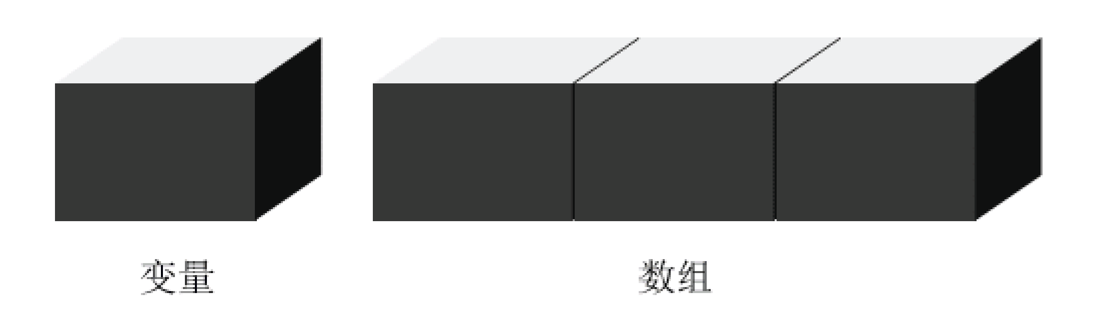


图 5.1 数组与变量的关系

数组中的每个实体都包含两项：键和值。其中键可以是数字、字符串或者数字和字符串的组合，用于标识数组中相应的值；而值被称为数组中的元素，可以定义为任意数据类型，甚至是混合类型。最终通过键来获取相应的值。例如，一个足球队通常会有几十个人，但是认识他们时首先会把他们看作是某队的成员，然后再通过他们的号码来区分每一名队员，这时，球队就是一个数组，而号码就是数组的下标（键），当指明是几号队员时就找到了这名队员（值）。

PHP 数组比许多其他高级语言中的数组更加灵活，不但支持数字索引数组，而且支持以字符串或字符串、数字混合为键名的关联数组。而在其他高级语言，如 Java 或者 C++等语言中，只支持数字索引数组。PHP 数组结构与其他语言数组结构的比较如图 5.2 和图 5.3 所示。

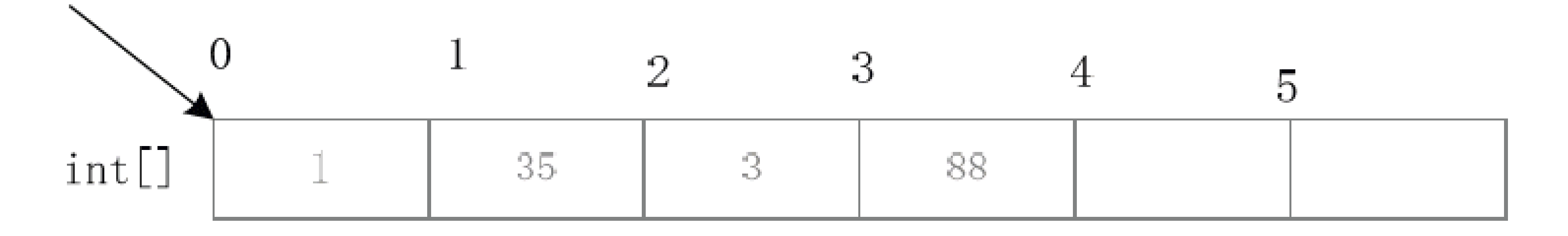


图 5.2 C++或者 Java 的数组结构

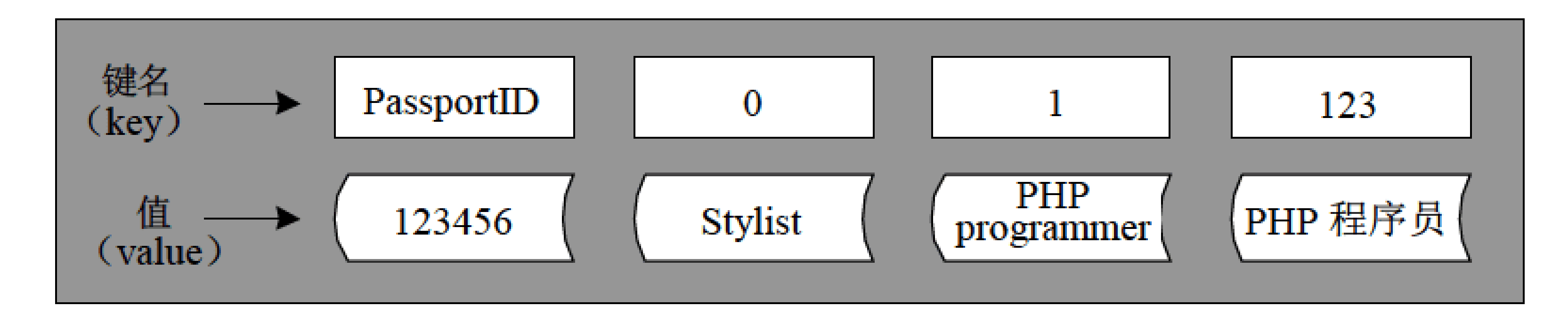


图 5.3 PHP 的数组结构

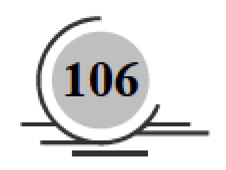
5.2 声 明 数 组

视频讲解：光盘\TM\Video\第 5 章\声明数组.exe

PHP 中声明数组的方式主要有两种：一种是应用数组函数声明数组；另一种是通过为数组元素赋值的方式声明数组。

5.2.1 数组命名规则

PHP 中声明数组的规则：数组的名称由一个“$”（美元符号）开始，第一个字符是字母或下划线，其



后是任意数量的字母、数字或下划线。例如：

```
$array_name=array('PHP'=>'php','ASP'=>'asp','JAVA'=>'java'); //以字符串作为数组索引，指定关键字
$_aa=array('PHP','Java',C#','Vb');                           //以数字作为数组索引，从 0 开始，没有指定关键字
$array[]=value1;
```

上述都是符合命名规则的数组，而下面的这两个数组则不符合命名规则。

```
//$1="array(1=>'a',2=>'b',3=>'c',4=>'d')";                //不可以以数字开头
$@=array('PHP'=>'php','JAVA'=>'java','JSP'=>'jsp');       //不可以使用特殊字符
```

在同一个程序中，标量变量和数组变量都不能重名。例如，如果已经存在一个名称为$string 的变量，而又创建一个名称为$string 的数组，那么前一个变量就会被覆盖。

数组的名称是区分大小写的，如$String 与$string 是不同的。

5.2.2　通过 PHP 函数创建数组

在 PHP 中，可以创建数组的函数如表 5.1 所示。这里主要讲解通过 array()函数创建数组。

表 5.1　创建数组的函数

函　　数	功　　能
array()	创建一个数组（创建数组最常用的函数）
array_combine()	以一个数组为关键字、另一个数组为值建立新数组
array_fill()	用值填充一个数组
array_pad()	用一个值把数组填充到指定长度
compact()	创建包含变量及其值的数组
range()	创建一个数组包含一定范围内的元素

应用 array()函数声明数组的方式如下：

```
array array ( [mixed ...])
```

参数 mixed 的语法为“key => value”，多个参数 mixed 用逗号分开，分别定义索引和值。索引可以是字符串或数字。如果省略了索引，会自动产生从 0 开始的整数索引。如果索引是整数，则下一个产生的索引将是目前最大的整数索引+1。如果定义了两个完全一样的索引，则后面一个会覆盖前一个。数组中各数据元素的数据类型可以不同，也可以是数组类型。当 mixed 是数组类型时，就是二维数组。

应用 array()函数声明数组时，数组下标既可以是数值索引也可以是关联索引。下标与数组元素值之间用“=>”进行连接，不同数组元素之间用逗号进行分隔。

应用 array()函数定义数组比较灵活，可以在函数体中只给出数组元素值，而不必给出键值。

例 5.1　通过 array()函数声明数组，并输出数组中的值。（**实例位置：光盘\TM\Instance\05\5.1**）

代码如下：

```
<?php
$array_name=array('PHP'=>'php','ASP'=>'jasp','C#'=>'c#');    //以字符串作为数组索引，指定关键字
print_r($array_name);                                        //输出数组
echo "<br>";                                                 //换行
echo $array_name[PHP];                                       //输出数组中的索引为 PHP 的元素
echo "<br>";
$_aa=array('PHP','Asp','C#','Vb');                           //以数字作为数组索引，从 0 开始，没有指定关键字
print_r($_aa);                                               //输出整个数组
echo "<br>";
echo $_aa[1];                                                //输出数组中的第一个元素
?>
```

运行结果如图 5.4 所示。

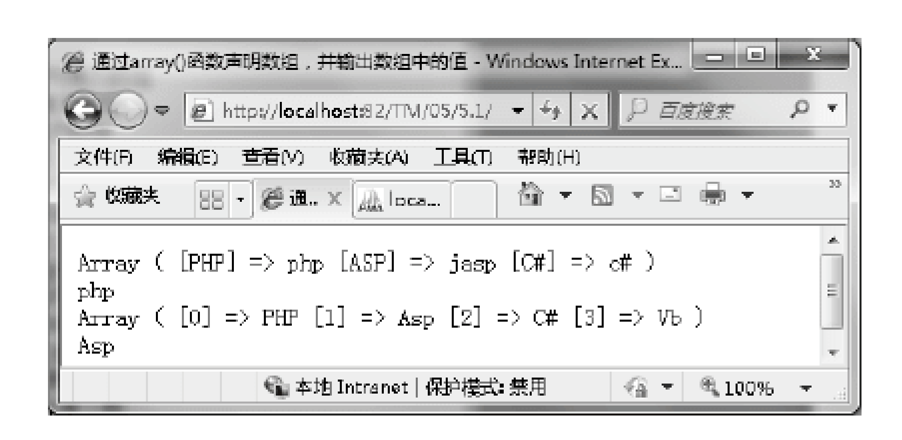


图 5.4 通过 array()函数声明数组

注意 可以通过给变量赋予一个没有参数的 array()来创建空数组，然后通过使用方括号（[]）来添加值。

PHP 提供创建数组的 array()语言结构。在使用其中的数据时，可以直接应用它们在数组中的排列顺序取值，该顺序称为数组的下标。例如：

```
<?php
    echo $_aa[ 0 ];                          //输出数组元素的第 0 个下标值
?>
```

结果为：PHP。

注意 应用这种方式定义数组时，下标默认是从 0 开始的，而不是 1，然后依次增加 1。所以为 0 的元素是指数组的第 1 个元素。

5.2.3 通过数组标识符“[]”创建数组

PHP 中另一种比较灵活的数组声明方式是通过数组标识符“[]”直接为数组元素赋值。如果在创建数组时不知所创建数组的大小，或在实际编写程序时数组的大小可能发生改变，采用这种数组创建的方法较好。

例 5.2 通过数组标识符“[]”创建数组。**（实例位置：光盘\TM\Instance\05\5.2）**

代码如下：

```
<?php
$array[0]="PHP";
$array[1]="编";
$array[2]="程";
$array[3]="词";
$array[4]="典";
print_r($array);                          //输出所创建数组的结构
?>
```

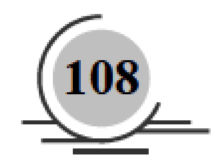
结果为：Array ([0] => PHP [1] => 编 [2] => 程 [3] => 词 [4] => 典)。

注意 通过直接为数组元素赋值的方式声明数组时，要求同一数组元素中的数组名相同。

5.3 数组的类型

视频讲解：光盘\TM\Video\第 5 章\数组的类型.exe

PHP 支持两种数组：数字索引数组（indexed array）和关联数组（associative array），数字索引数组使用数字作为键，关联数组使用字符串作为键。



5.3.1　数字索引数组

PHP 数字索引由数字组成，下标从 0 开始，数字索引一般表示数组元素在数组中的位置。数字索引数组默认索引值从数字 0 开始，不需要特别指定，PHP 会自动为索引数组的键名赋一个整数值，然后从这个值开始自动增加，当然，也可以指定从某个位置开始保存数据。

数字索引数组可以构造成一系列键-值(key-value)对，其中每一对都是数组的一个项目或元素(element)。对于列表中的每个项目，都有一个与之关联的键（key）（或索引（index）），如图 5.5 所示。

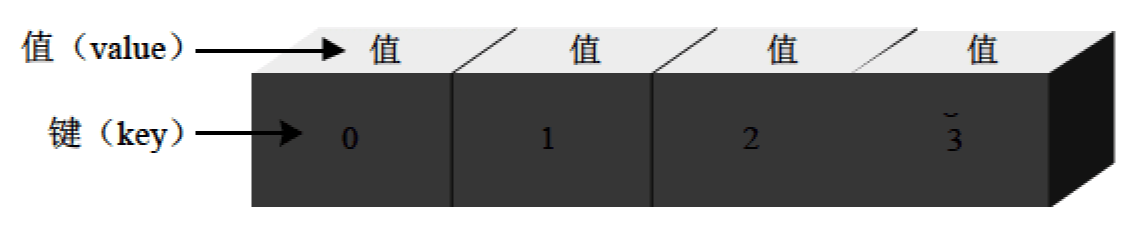


图 5.5　数字索引数组

5.3.2　关联数组

关联数组的键名可以是数值和字符串混合的形式，而不像数字索引数组的键名只能为数字，在一个数组中，只要键名中有一个不是数字，那么这个数组就叫做关联数组。

关联数组使用字符串索引（或键）来访问存储在数组中的值，如图 5.6 所示。关联索引的数组对于数据库层交互非常有用。

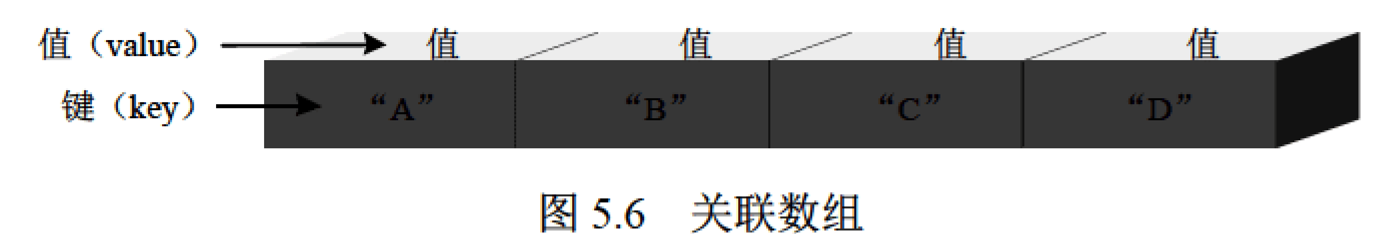


图 5.6　关联数组

例 5.3　创建一个关联数组。**（实例位置：光盘\TM\Instance\05\5.3）**

代码如下：

```
<?php
$array = array("first"=>"PHP","second"=>"ASP","third"=>"WEB");
echo $array["second"];                    //输出索引为 second 的元素的值
$array["third"]=" JAVA";                  //为索引为 third 的元素重新赋值
echo $array["third"];                     //输出索引为 third 的元素的值
?>
```

结果为：ASP JAVA。

技巧 关联数组的键名可以是任何一个整数或字符串。如果键名是一个字符串，不要忘记给键名或索引加上定界修饰符：单引号（'）或双引号（"）。对于数字索引数组，为了避免不必要的麻烦，最好也加上定界符。

5.4　输 出 数 组

视频讲解：光盘\TM\Video\第 5 章\输出数组.exe

PHP 中对数组元素进行输出可以通过输出语句来实现，如 echo、print 语句等，但应用这种输出方式只

能对某数组中某一元素进行输出。而通过 print_r()和 var_dump()函数可以将数组结构进行输出。

print_r()函数的语法如下：

```
bool print_r ( mixed expression )
```

如果该函数的参数 expression 为普通的整型、字符型或实型变量，则输出该变量本身；如果该参数为数组，则按键值和元素的顺序显示出该数组中的所有元素。

例 5.4 使用 print_r()函数输出数组的结构。（**实例位置：光盘\TM\Instance\05\5.4**）

代码如下：

```
<?php
    $array=array(1=>"PHP 编程词典",2=>"C#编程词典",3=>"JAVA 编程词典");
    print_r($array);
?>
```

结果为：Array ([1] => PHP 编程词典 [2] => C#编程词典 [3] => JAVA 编程词典)。

var_dump()函数可以输出数组（或对象）、元素数量以及每个字符串的长度，还能够以缩进方式输出数组或对象的结构。

语法：

```
void var_dump( mixed expression [,mixed expression [,…]])
```

例 5.5 通过 var_dump()函数输出数组的结构。（**实例位置：光盘\TM\Instance\05\5.5**）

代码如下：

```
<?php
$array=array("PHP 开发实战宝典","PHP 从入门到精通","学通 PHP 的 24 堂课 ");
var_dump($array);
$arrays=array('first'=>" PHP 开发实战宝典",'second'=>"PHP 从入门到精通",'third'=>"学通 PHP 的 24 堂课");
var_dump($arrays);
?>
```

运行结果如图 5.7 所示。



图 5.7　通过 var_dump()函数输出数组的结构

5.5　数组的构造

视频讲解：光盘\TM\Video\第 5 章\数组的构造.exe

5.5.1　创建一维数组

当一个数组的元素是变量时，称这个数组为一维数组。一维数组是最普通的数组，它只保存一列内容。

例 5.6 创建两个一维数组，一个是数字索引数组，另一个是关联数组。（**实例位置：光盘\TM\Instance\05\5.6**）

代码如下：

```
<?php
```



```
$array=array("PHP 开发实战宝典","PHP 从入门到精通","PHP 求职宝典");
print_r($array);
$arrays=array('first'=>" PHP 开发实战宝典",'second'=>"PHP 从入门到精通",'third'=>" PHP 求职宝典");
print_r($arrays);
?>
```

运行结果如图 5.8 所示。

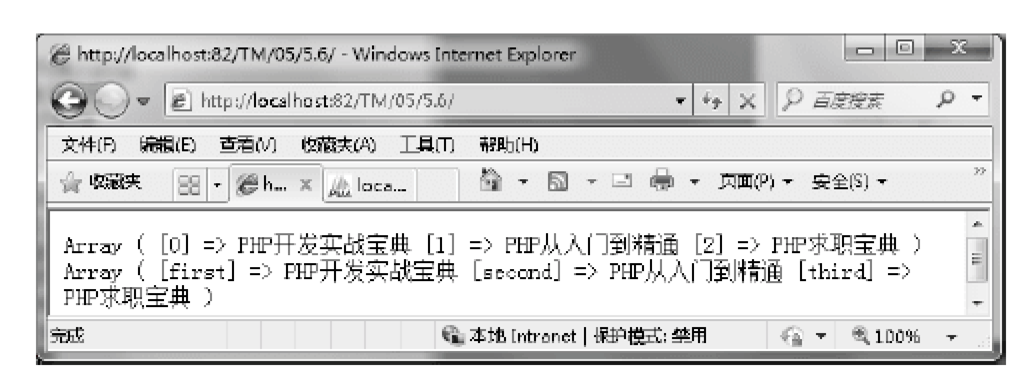


图 5.8　创建并输出一维数组

5.5.2　创建二维数组

一个数组的元素如果是一维数组，则称这个数组是二维数组。二维数组的定义和使用与一维数组基本相同，唯一的区别是二维数组的元素仍然是数组。

例 5.7　声明一个二维数组。（**实例位置：光盘\TM\Instance\05\5.7**）

代码如下：

```
<?php
    $str = array (
        "PHP 类图书"=>array ("PHP 求职宝典","PHP 经典编程","PHP 开发实战宝典"),
        "JAVA 类图书"=>array ("a"=>"JAVA 范例手册","b"=>"JAVA WEB 范例宝典"),
        "ASP 类图书"=>array ("ASP 学习手册",2=>"ASP 范例宝典","ASP 开发实战宝典")
    );                                          //声明数组
    print_r ( $str) ;                           //输出数组元素
?>
```

结果为：

```
Array(
  [PHP 类图书] => Array(
     [0] => PHP 求职宝典
     [1] => PHP 经典编程
     [2] => PHP 开发实战宝典
  )
   [JAVA 类图书] => Array(
     [a] => JAVA 范例手册
     [b] => JAVA WEB 范例宝典
  )
   [ASP 类图书] => Array(
     [0] => ASP 学习手册
     [2] => ASP 范例宝典
     [3] => ASP 开发实战宝典
  )
)。
```

上面的代码实现一个二维数组的声明，按照同样的思路可以创建更高维数的数组，如三维数组或四维数组等。

5.6　遍 历 数 组

视频讲解：光盘\TM\Video\第 5 章\遍历数组.exe

遍历数组中的所有元素是常用的一种操作，在遍历的过程中可以完成查询或其他功能。如去商场购物，如果想要买一台电脑，就需要在商场中逛一遍，看是否有满意的机型，而逛商场的过程就相当于遍历数组的操作。在 PHP 中遍历数组的方法有多种。下面介绍最常用的 3 种方法。

5.6.1 foreach 结构遍历数组

遍历数组元素最常用的方法是使用 foreach 结构。foreach 结构并非操作数组本身，而是操作数组的一个备份。

例 5.8 通过 foreach 结构遍历数组中的数据。**（实例位置：光盘\TM\Instance\05\5.8）**

代码如下：

```
<?php
	$name=array('编程词典网','编程体验网','编程资源网');                                                                   //声明数组
	$url = array('0'=>'www.mrbccd.com','1'=>'www.mingribook.com','2'=>'www.mingrisoft.com',);   //声明数组
	foreach ($name as $key=>$value) {                                                    //遍历数组
		echo $value."    ——".$url[$key].'<br><br>';                                     //输出数组中的数据
	}
?>
```

运行结果如图 5.9 所示。

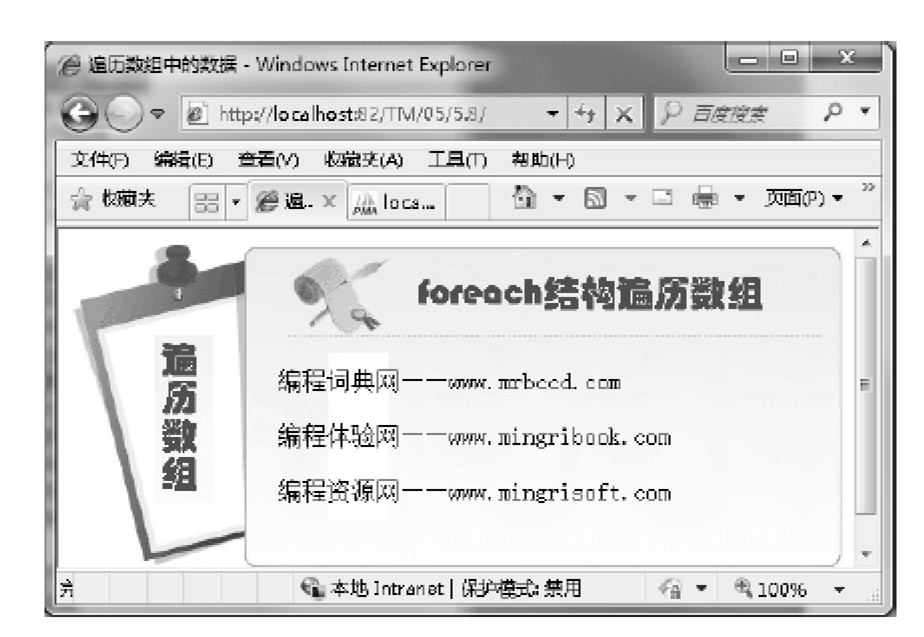


图 5.9 通过 foreach 结构遍历数组中的数据

5.6.2 list()函数遍历数组

list()函数把数组中的值赋给一些变量。与 array()函数类似，list()函数不是真正的函数，而是一种语言结构。list()函数仅能用于数字索引且索引值从 0 开始的数组，其语法格式如下：

```
void list ( mixed ...)
```

参数 mixed 为被赋值的变量名称。

例 5.9 应用 each()和 list()函数，输出存储在数组中的用户登录信息。**（实例位置：光盘\TM\Instance\05\5.9）**

具体开发步骤如下。

（1）应用开发工具（如 Dreamweaver），新建一个 PHP 动态页，命名为 index.php。

（2）应用 HTML 标记设计页面，首先建立用户登录表单，用于提交用户登录信息，然后应用 each()函数提取全局数组$_POST 中的内容，并最终应用 while 循环输出用户所提交的注册信息。代码如下：

```
<!-- --------------------------------------------------------定义用户登录表单---------------------------------------------------- -->
<form name="form1" method="post">
	<table width="233" border="0" cellpadding="0" cellspacing="0">
		<tr>
			<td width="62" height="24" align="right" bgcolor="#FFFFFF"><span class="STYLE3">用户名：</span></td>
			<td width="189" height="24" bgcolor="#FFFFFF"><input name="username" type="text" id="user" size= "18"></td>
		</tr>
		<tr>
			<td height="24" align="right" bgcolor="#FFFFFF"><span class="STYLE3">密  码：</span></td>
			<td height="24" bgcolor="#FFFFFF"><input name="pwd" type="password" id="pwd" size="18"></td>
		</tr>
		<tr align="center" bgcolor="#CCFF33">
			<td height="24" colspan="2" bgcolor="#FFFFFF">
				<input type="image" name="imageField" src="images/bg1.JPG">  
				<input type="image" name="imageField2" src="images/bg2.JPG" onClick="form.reset();return false;"></td>
```

```
            </tr>
        </table>
</form>
<?php
        while(list($name,$value)=each($_POST)){                //输出用户登录信息
            if($name!="imageField_x" and $name!="imageField_y" ){
                echo "$name=$value<br>";
            }
        }
?>
```

（3）在 IE 浏览器中输入地址，按 Enter 键，输入用户名及密码，单击“提交”按钮。运行结果如图 5.10 所示。

说明 each()函数用于返回当前指针位置的数组值，并将指针推进一个位置。返回的数组包含 4 个键，键 0 和 key 包含键名，而键 1 和 value 包含相应的数据。如果程序在执行 each()函数时，指针已经位于数组末尾，则返回 false。

5.6.3　for 语句遍历数组

想要通过 for 循环遍历数组，首先必须应用 count()函数获取数组中的单元数目，然后将数组中的单元数目作为 for 循环的条件，执行循环输出。

count()函数的语法如下：

```
int count ( mixed array [, int mode])
```

参数 array 为指定输入的数组；参数 mode 为可选参数，若设为 COUNT_RECURSIVE（或 1），将递归地对数组计数。这对计算多维数组的所有单元尤其有用。此参数的默认值为 0。

有关 for 循环语句的详细讲解可参考第 3 章流程控制语句的 3.3.3 节，这里不再赘述。

例 5.10　本实例应用 count()函数和 for 语句循环输出数组中的数据。（**实例位置：光盘\TM\Instance\05\5.10**）

代码如下：

```
<?php
    $name=array('编程词典网','编程体验网','编程资源网');                    //声明数组
    $url = array('0'=>'www.mrbccd.com','1'=>'www. mingribook.com','2'=>'www.mingrisoft.com',);   //声明数组
    for($i=0; $i<count($name);$i++) {                    //根据数组中元素个数，循环遍历数组
        echo $name[0]."    ——".$url[0].'<br><br>';      //输出数组中的数据
    }
?>
```

运行结果如图 5.11 所示。



图 5.10　应用 list()函数获取用户登录信息



图 5.11　应用 count()函数和 for 语句循环输出数组中的数据

5.7 PHP 全局数组

应用 PHP 提供的全局数组，可以获取大量与环境有关的信息。例如，可以应用这些数组获取当前用户会话、用户操作环境和本地操作环境等信息。下面对 PHP 中常用的全局数组进行介绍。

5.7.1 $_SERVER[]全局数组

$_SERVER[]全局数组包含由 Web 服务器创建的信息，应用该数组可以获取服务器和客户配置及当前请求的有关信息。下面对$_SERVER[]数组进行介绍，如表 5.2 所示。

表 5.2 $_SERVER[]全局数组

数 组 元 素	说 明
$_SERVER['SERVER_ADDR']	当前运行脚本所在服务器的 IP 地址
$_SERVER['SERVER_NAME']	当前运行脚本所在服务器主机的名称。如果该脚本运行在一个虚拟主机上，该名称由虚拟主机所设置的值决定
$_SERVER['REQUEST_METHOD']	访问页面时的请求方法，如 GET、HEAD、POST、PUT。如果请求的方式是 HEAD，PHP 脚本将在送出头信息后中止（这意味着在产生任何输出后，不再有输出缓冲）
$_SERVER['REMOTE_ADDR']	正在浏览当前页面用户的 IP 地址
$_SERVER['REMOTE_HOST']	正在浏览当前页面用户的主机名。反向域名解析基于该用户的 REMOTE_ADDR
$_SERVER['REMOTE_PORT']	用户连接到服务器时所使用的端口
$_SERVER['SCRIPT_FILENAME']	当前执行脚本的绝对路径名。注意：如果脚本在 CLI 中被执行，作为相对路径，如 file.php 或者../file.php，$_SERVER['SCRIPT_FILENAME']将包含用户指定的相对路径
$_SERVER['SERVER_PORT']	服务器所使用的端口，默认为 80。如果使用 SSL 安全连接，则该值为用户设置的 HTTP 端口
$_SERVER['SERVER_SIGNATURE']	包含服务器版本和虚拟主机名的字符串
$_SERVER['DOCUMENT_ROOT']	当前运行脚本所在的文档根目录。在服务器配置文件中定义

例 5.11 通过$_SERVER[]全局数组获取服务器和客户端的 IP 地址，客户端连接主机的端口号和服务器的根目录。**（实例位置：光盘\TM\Instance\05\5.11）**

其代码如下。

```
<?php
	echo "当前服务器 IP 地址是：<b>".$_SERVER['SERVER_ADDR']."</b><br>";
	echo "当前服务器的主机名称是：<b>".$_SERVER['SERVER_NAME']."</b><br>";
	echo "客户端 IP 地址是：<b>".$_SERVER['REMOTE_ADDR']."</b><br>";
	echo "客户端连接到主机所使用的端口：<b>".$_SERVER['REMOTE_PORT']."</b><br>";
	echo "当前运行的脚本所在文档的根目录：<b>".$_SERVER['DOCUMENT_ROOT']."</b><br>";
?>
```



运行结果如图 5.12 所示。

当前服务器IP地址是：127.0.0.1
当前服务器的主机名称是：localhost
客户端IP地址是：127.0.0.1
客户端连接到主机所使用的端口：9557
当前运行的脚本所在文档的根目录：E:/AppServ/www

图 5.12 获取服务器和客户端的 IP 地址

5.7.2 $_GET[]和$_POST[]全局数组

PHP 中提供$_GET[]和$_POST[]全局数组，分别用来接收 GET 和 POST 方法传递到当前页面的数据。

例 5.12 下面开发一个实例，获取用户的登录信息。分别通过 GET 和 POST 方法完成数据的提交，并且应用$_GET[]和$_POST[]全局数组获取用户提交的数据，从返回的结果中体会二者之间的区别。**（实例位置：光盘\TM\Instance\05\5.12）**

具体实现步骤如下：

（1）创建 index.php 文件，同时定义两个 form 表单，分别使用 GET 和 POST 方法提交数据，将通过 GET 方法提交的数据传递到 get.php 文件，将通过 POST 方法提交的数据传递到 post.php 文件。

（2）创建 get.php 文件，通过$_GET[]全局数组获取 GET 方法提交的数据。运行结果如图 5.13 所示。

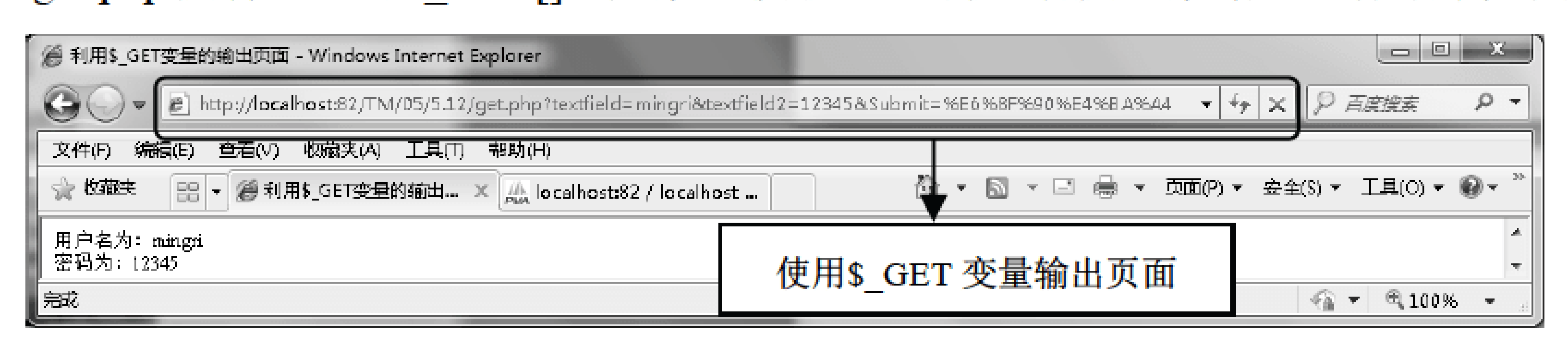


图 5.13 利用$_GET 变量的输出页面

具体代码如下：

```
<?php
if(isset($_GET['Submit']) and $_GET['Submit']=="提交"){
	echo "用户名为：".$_GET['textfield']."<br>";
	echo "密码为：".$_GET['textfield2'];
}
?>
```

（3）创建 post.php 文件，通过$_POST[]全局数组获取 POST 方法提交的数据。运行结果如图 5.14 所示。

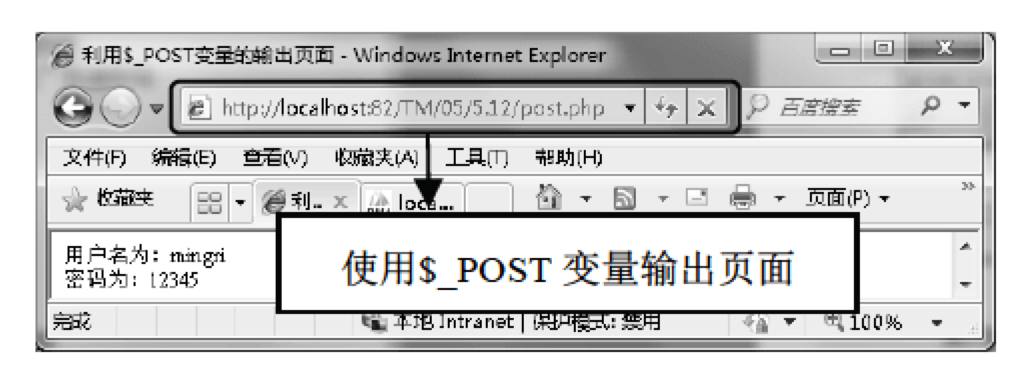


图 5.14 利用$_POST 变量的输出页面

具体代码如下：

```
<?php
if(isset($_POST['Submit2']) and $_POST['Submit2']=="提交"){
	echo "用户名为：".$_POST['textfield3']."<br>";
	echo "密码为：".$_POST['textfield22'];
}
?>
```

5.7.3 $_COOKIE 全局数组

$_COOKIE[]全局数组存储通过 http Cookie 传递到脚本的信息。PHP 中可以通过 setcookie()函数设置

Cookie 的值，用$_COOKIE[]数组接收 Cookie 的值，$_COOKIE[]数组的下标为 Cookie 的名称。

例如，通过 setCookie()函数创建一个 Cookie，并通过$_COOKIE[]全局数组获取 Cookie 的值。其代码如下：

```
<?php
setCookie("mingri",'明日科技');
setCookie("mingri", '明日科技', time()+60);                //设置 Cookie 有效时间为 60 秒
echo "读取 Cookie：".$_COOKIE['mingri'];                   //通过$_COOKIE[]读取 Cookie 的值
?>
```

5.7.4 $_ENV[]全局数组

$_ENV[]全局数组用于提供与服务器有关的信息。例如，

- ☑ $_ENV["HOSTNAME"]：获取服务器名称。
- ☑ $_ENV["SHELL"]：获取系统 shell。

5.7.5 $_REQUEST[]全局数组

可以用$_REQUEST[]全局数组获取 GET 方法、POST 方法和 http Cookie 传递到脚本的信息。如果在编写程序时，不能确定是通过什么方法提交的数据，那么就可以通过$_REQUEST[]全局数组获取提交到当前页面的数据。

5.7.6 $_SESSION[]全局数组

$_SESSION[]全局数组用于获取会话变量的相关信息。

例如，初始化 SESSION 变量，通过$_SESSION[]全局数组为 SESSION 变量赋值，最后通过$_SESSION[]全局数组输出 SESSION 的值。其代码如下：

```
<?php
session_start();                                          //初始化 SESSION 变量
$_SESSION['mr']="mingri";                                 //为 SESSION 变量赋值
echo   $_SESSION['mr']                                    //输出 SESSION 变量的值
?>
```

5.7.7 $_FILES[]全局数组

与其他全局数组不同，$_FILES[]全局数组为一个多维数组，该数组用于获取通过 POST 方法上传文件时的相关信息。如果为单文件上传，则该数组为二维数组，如果为多文件上传，则该数组为三维数组。下面对该数组的具体参数取值进行描述。

- ☑ $_FILES["file"]["name"]：从客户端上传的文件名称。
- ☑ $_FILES["file"]["type"]：从客户端上传的文件类型。
- ☑ $_FILES["userfile"]["size"]：已上传文件的大小。
- ☑ $_FILES["file"]["tmp_name"]：文件上传到服务器后，在服务器中的临时文件名。
- ☑ $_FILES["file"]["error"]：返回在上传过程中发生错误的错误代号。

5.8　PHP 的数组函数

下面介绍一些在实际程序开发中比较常用的数组函数，如果需要使用数组函数实现某些特殊的功能，可以参考 PHP 中文手册，其中有对所有数组函数的详细介绍，可以根据所要实现的功能进行选择、学习或者研究。

5.8.1　统计数组元素个数

视频讲解：光盘\TM\Video\第 5 章\统计数组元素个数.exe

在 PHP 中，应用 count()函数可以对数组中的元素个数进行统计，在讲解使用 for 循环遍历数组时已经应用到，下面详细介绍一下该函数。语法格式如下：

```
int count ( mixed var [, int mode])
```

参数 var 指定操作的数组对象；参数 mode 为可选参数，默认值是 0。如果 mode 的值设置为 COUNT_RECURSIVE（或 1），count()函数检测多维数组。该函数返回数组元素的个数。

说明 如果 count()函数的操作对象是 NULL，那么返回结果是 0。count()函数对没有初始化的变量返回 0，但对于空的数组也会返回 0。如果要判断变量是否初始化，则可以应用 isset()函数。count()函数不能识别无限递归。

例 5.13　下面使用 count()函数统计数组中元素个数，并输出统计结果。**（实例位置：光盘\TM\Instance\05\5.13）**

代码如下：

```
<?php
$array=array(0 =>'PHP 开发实战宝典', 1 =>'JAVA 学习手册', 2 =>'HTML 从入门到精通');
echo count($array);                              //统计数组中元素个数，并使用 echo 语句输出统计结果
?>
```

运行结果为：3。

5.8.2　向数组中添加元素

在 PHP 中，使用 array_push()函数可以向数组中添加元素，该函数将传入的元素添加到某个数组的末尾，并返回数组新的单元总数。语法如下：

```
int array_push ( array array, mixed var [, mixed ...])
```

参数 array 为指定的数组；参数 var 是压入数组中的值。

例 5.14　下面使用 array_push()函数向数组中添加元素，并输出添加元素后的数组。**（实例位置：光盘\TM\Instance\05\5.14）**

代码如下：

```
<?php
$array=array(0 =>'PHP 求职宝典', 1 =>'JAVA 范例宝典');                    //声明数组
echo "添加前的数组元素：";
print_r($array);
echo "<br>";
```

```
array_push($array,'VB 标准教程','VC 从入门到精通');                    //向数组中添加元素
echo "添加后的数组元素：";
print_r($array);                                                        //输出添加后的数组结构
?>
```

运行结果如图 5.15 所示。

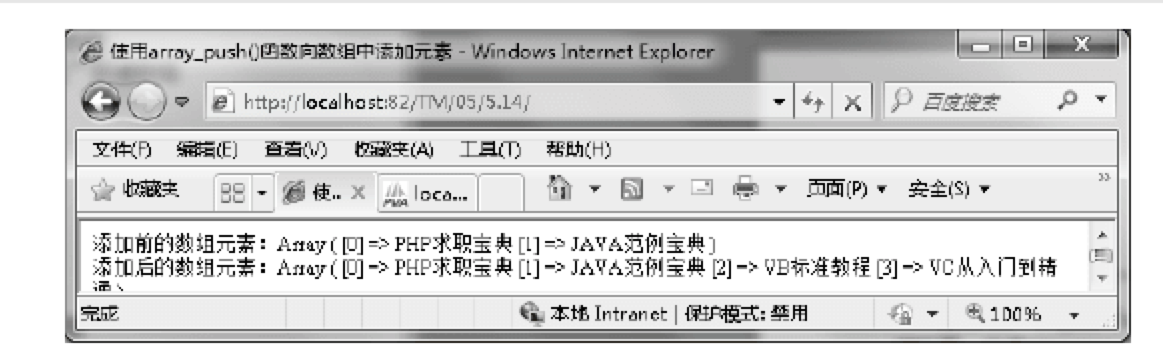


图 5.15　使用 array_push()函数向数组中添加元素

5.8.3　获取数组中最后一个元素

在 PHP 中，通过 array_pop()函数可以获取并返回数组中的最后一个元素，并将数组的长度减 1，如果数组为空（或者不是数组）将返回 NULL。语法格式如下：

```
mixed array_pop ( array array)
```

参数 array 为输入的数组。

例 5.15　首先应用 array_push()函数向数组中添加元素，然后应用 array_pop()函数获取数组中最后一个元素，最后输出最后一个元素值。**（实例位置：光盘\TM\Instance\05\5.15）**

代码如下：

```
<?php
$array=array(0 =>'PHP 从入门到精通', 1 =>'JAVA 从入门到精通');        //声明数组
array_push($array,'VB 开发实战宝典','VC 开发实战宝典');                //向数组中添加元素
$last_array=array_pop($array);                                         //获取数组中最后一个元素
echo $last_array;                                                      //输出最后一个元素值
?>
```

运行结果为：VC 开发实战宝典。

5.8.4　删除数组中重复元素

array_unique()函数将数组元素的值作为字符串排序，然后对每个值只保留第一个键名，忽略所有后面的键名，即删除数组中重复的元素。

语法如下：

```
array array_unique ( array array)
```

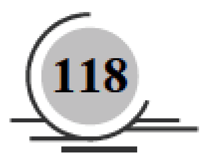
参数 array 为输入的数组。

说明　虽然 array_unique()函数只保留重复值的第一个键名。但是，该键名并不是在未排序的数组中同一个值的第一个出现的键名，只有当两个字符串的表达式完全相同时（(string) $elem1 === (string) $elem2），第一个单元才被保留。

例 5.16　首先定义一个数组，然后应用 array_push()函数向数组中添加元素，并输出数组，最后应用 array_unique()函数，删除数组中重复元素，并输出数组。**（实例位置：光盘\TM\Instance\05\5.16）**

代码如下：

```
<?php
$arr_int = array ("PHP", ".NET","ASP");                                //定义数组
array_push ($arr_int, "PHP","ASP");                                    //向数组中添加元素
print_r($arr_int);                                                     //输出添加后的数组
$result=array_unique($arr_int);                                        //删除添加后数组中重复的元素
print_r($result);                                                      //输出删除重复元素后的数组
?>
```



运行结果如图 5.16 所示。

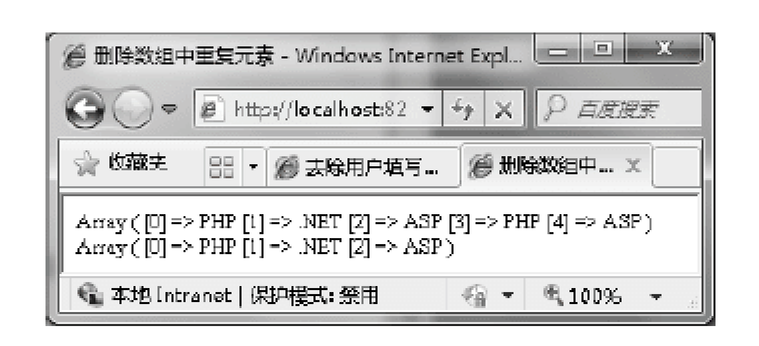


图 5.16　删除数组中重复元素

说明 使用 unset()函数可删除数组中的某个元素，例如，将例 5.16 中$arr_int 数组的第 2 个元素删除，关键代码如下：

```
unset($arr_int[1]);
```

5.8.5　获取数组中指定元素的键名

获取数组中指定元素的键名主要是通过数组函数来实现，本节分别介绍 array_search()函数和 array_keys()函数获取数组中重复元素的所有键名。

☑　使用 array_search()函数可获取数组中指定元素的键名

使用 array_search()函数可获取数组中指定元素的键名，在数组中搜索给定的值，找到后返回键名，否则返回 false。其语法如下：

```
mixed array_search ( mixed needle, array haystack [, bool strict])
```

array_search()函数的参数说明如表 5.3 所示。

表 5.3　array_search()函数的参数说明

参　　数	说　　明
needle	指定在数组中搜索的值，如果 needle 是字符串，则比较以区分大小写的方式进行
haystack	指定被搜索的数组
strict	可选参数，如果值为 true，还将在 haystack 中检查 needle 的类型

说明 array_search()函数是区分字母大小写的。

例 5.17　下面使用 array_search()函数获取数组中元素的键名。**（实例位置：光盘\TM\Instance\05\5.17）**

具体代码如下：

```
<?php
$arr=array("葡萄","山竹","橙子","西瓜");    //创建数组，数组中有 4 个元素
$name=array_search("西瓜",$arr);           //使用 array_search 获取$arr 数组中“西瓜”的键名，然后将获取的结果
                                             赋给$name 变量
echo $name;                                //输出结果
?>
```

运行结果为：3。

使用 array_keys()函数获取数组中重复元素的所有键名。

如果查询的元素在数组中出现两次以上，那么 array_keys()函数则返回第一个匹配的键名。如果想要返回所有匹配的键名，则需要使用 array_keys()函数。语法如下：

```
array array_keys ( array input [, mixed search_value [, bool strict]] )
```

array_keys()返回 input 数组中的数字或者字符串的键名。如果指定可选参数 search_value，则只返回该值的键名。否则 input 数组中的所有键名都会被返回。

例 5.18　下面使用 array_keys()函数来获取数组中重复元素的所有键名。**（实例位置：光盘\TM\Instance\05\5.18）**

具体代码如下：

```
<?php
$arr=array("葡萄","山竹","橙子","西瓜","山竹");
$name=array_keys($arr,"山竹");             //使用 array_keys()函数获取$arr 数组中“山竹”的所有键值
```

```
print_r($name);                         //因为 array_keys()函数返回的是数组类型的值，所以使用 print_r 输出
?>
```

运行结果为：Array ([0] => 1 [1] => 4)。

5.9 实　战

视频讲解：光盘\TM\Video\第 5 章\实战.exe

5.9.1 获取上传文件的数据

例 5.19 获取上传文件的数据，需要使用$_FILES[]全局数组中的$_FILES["file"]["name"]实现，本实例使用该全局数组获取上传文件的名称。（**实例位置：光盘\TM\Instance\05\5.19**）

具体代码如下：

```
<?php
if(isset($_POST['Submit']) and $_POST['Submit']=="获取上传文件名"){     //判断按钮的值是否存在，是否为空
     echo $_FILES["file"]["name"];                                    //输出上传文件名
}
?>
```

运行结果如图 5.17 所示。

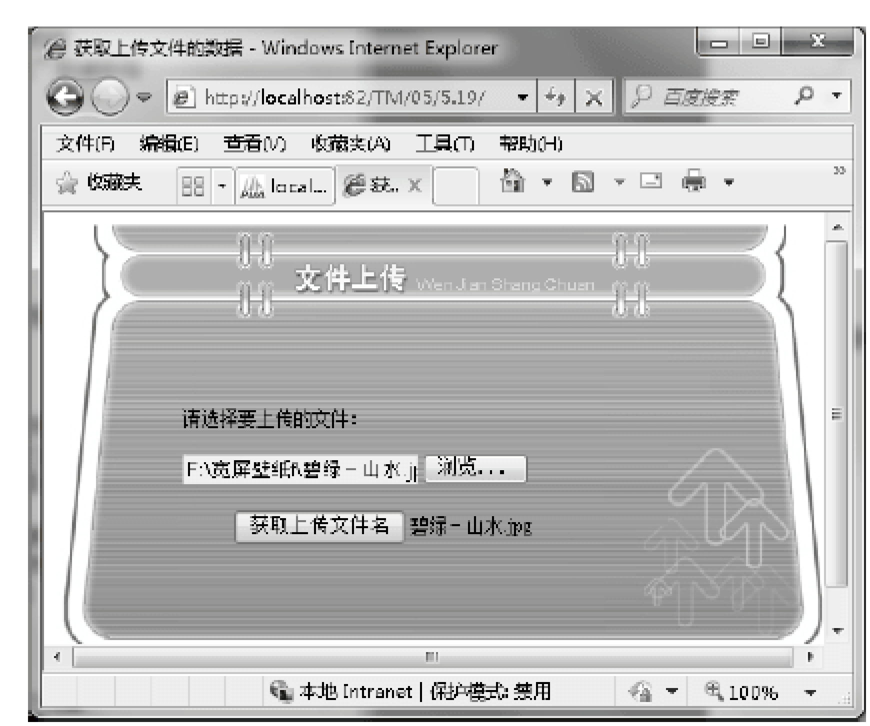


图 5.17　获取上传文件的数据

5.9.2 投票管理系统

例 5.20 在开发一个投票管理系统时，经常需要在后台添加投票选项到投票系统，作为投票的内容。下面应用 explode()函数对添加的投票选项通过“*”进行区分，然后通过 while 循环语句输出添加的投票选项。（**实例位置：光盘\TM\Instance\05\5.20**）

具体开发步骤如下。

（1）应用开发工具（如 Dreamweaver），新建一个 PHP 动态页，存储为 index.php。

（2）应用 HTML 标记设计页面，首先创建投票选项提交的表单，然后应用 each()函数提取全局数组$_POST 中的内容，最后应用 while 循环输出投票选项内容。其关键代码如下：

```
<?php
if($_POST[Submit]!=""){                              //判断提交的值不为空
          $content=$_POST[content];                  //获取提交的内容
          $data=explode("*",$content);               //通过 explode()函数以“*”分隔提交的内容
          while(list($name,$value)=each($data)){     //循环输出数组中的数据
               echo '<input type="checkbox" name="checkbox" value="checkbox">';
               echo $value."\n";
          }
     }
?>
```

（3）在 IE 浏览器中输入地址，按 Enter 键，输入投票选项的内容，各选项间用“*”进行分隔，单击“提交”按钮。运行结果如图 5.18 所示。

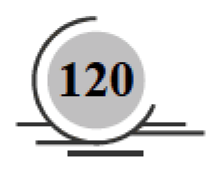




图 5.18　在投票管理系统中应用 explode()函数

5.9.3　获取用户注册信息

例 5.21　用户注册是大多数网站都具备的功能，本实例设计的用户注册信息包括用户名、密码、性别、学历、爱好、个人简介，信息填写完成后，单击“提交”按钮，将信息输出到页面中。（**实例位置：光盘\TM\Instance\05\5.21**）

代码如下：

```
<?php
echo "用户名：".$_POST['textfield']."<br>";                    //输出用户名
echo "密码：".$_POST['textfield2']."<br>";                     //输出密码
echo "性别：".$_POST['radiobutton']."<br>";                    //输出性别
echo "学历：".$_POST['select']."<br>";                         //输出学历
//下面为循环输出爱好信息
echo "爱好：";
$array=$_POST['checkbox'];
foreach($array as $arr){
    echo $arr."  ";
}
echo "<br>";
echo "个人简介：".$_POST['textarea'];                           //输出个人简介
?>
```

运行效果如图 5.19 所示。

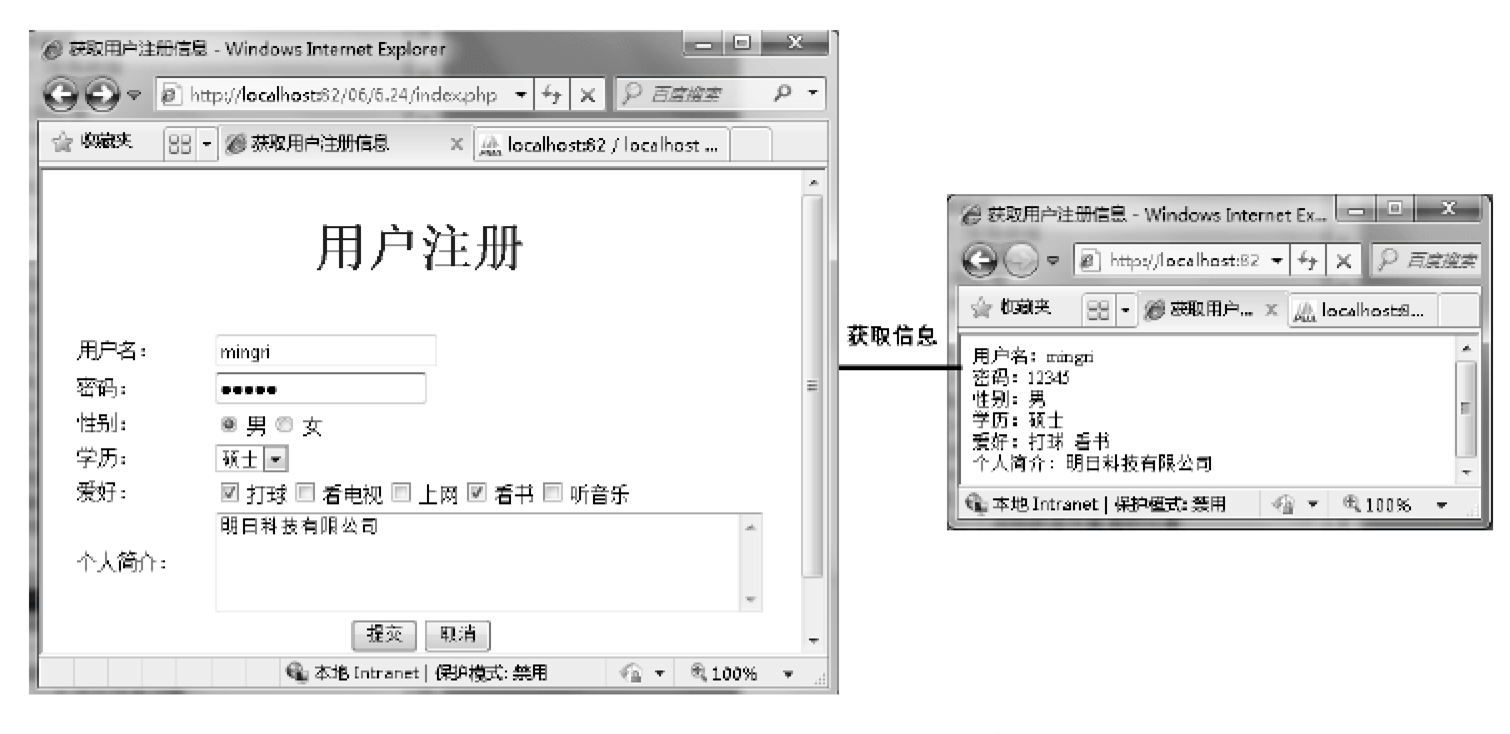


图 5.19　获取用户注册信息

5.9.4 车牌摇号

例 5.22 通过随机函数 rand()实现随机抽取数组中存储的车牌号码。代码如下：（**实例位置：光盘\TM\Instance\05\5.22**）

```
<?php
$numbers = array('00000', '11111', '22222', '33333', '66666', '88888');
Echo "车牌号码：";
print_r($numbers);
$rand = rand(0,6);
Echo "<br>随机取得车牌号码：".$numbers[$rand];
?>
```

运行效果如图 5.20 所示。

5.9.5 向数组中添加元素

例 5.23 使用 array_push()函数可实现向数组中添加元素，下面使用该函数向$arr 数组中添加两个元素。（**实例位置：光盘\TM\Instance\05\5.23**）

代码如下：

```
<?php
$arr=array("坚韧","创新");                    //创建数组
echo "原数组元素：";
print_r($arr);
echo "<br>";
array_push($arr,"博学","笃行");               //向数组中添加两个元素
echo "添加后的数组元素：";
print_r($arr);                                //输出添加元素后的数组
?>
```

运行结果如图 5.21 所示。

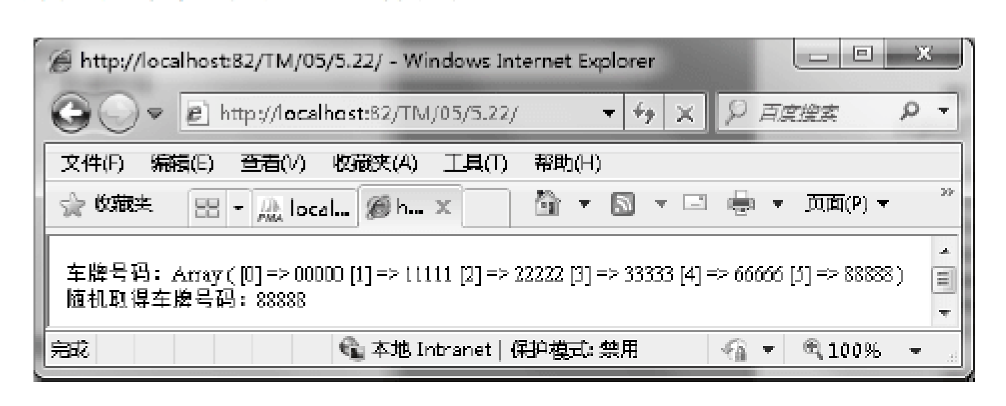


图 5.20 车牌摇号

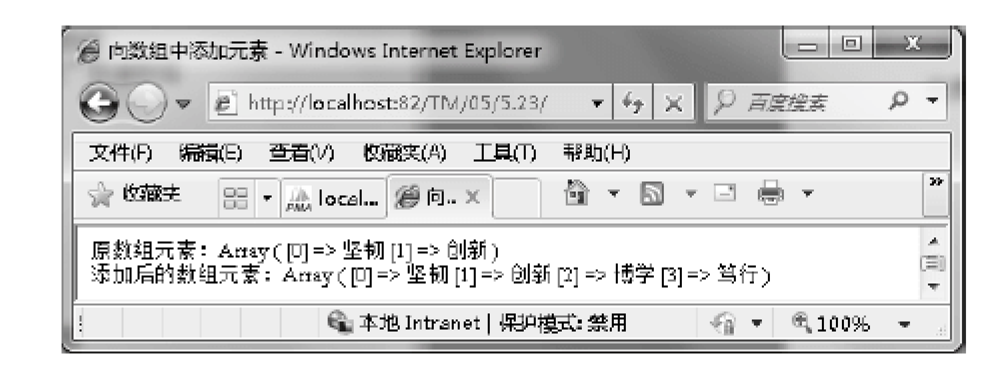


图 5.21 向数组中添加元素

5.10 小 结

本章对数组的讲解结合了笔者的实践经验，主要讲解了数组中常用的操作方法，包括遍历数组、统计数组元素个数、向数组中添加元素等，并且配合典型的实例，加深读者对数组函数的理解。还对数组的定义、类型和构造进行了介绍。注意，这里介绍的只是 PHP 数组中的部分函数，有关 PHP 数组中其他函数的介绍，读者可以参考 PHP 中文手册。



5.11　学习成果检验

1．尝试声明一个一维数组和一个二维数组，并对数组元素进行输出。**（答案位置：光盘\TM\Instance\05\5.24）**

2．尝试开发一个页面，应用 explode()函数以“*”为分隔符，将字符串转换成数组，然后合成一个新数组，并从新数组中随机抽取 3 个元素值，最后输出随机抽取的元素，运行结果如图 5.22 所示。**（答案位置：光盘\TM\Instance\05\5.25）**

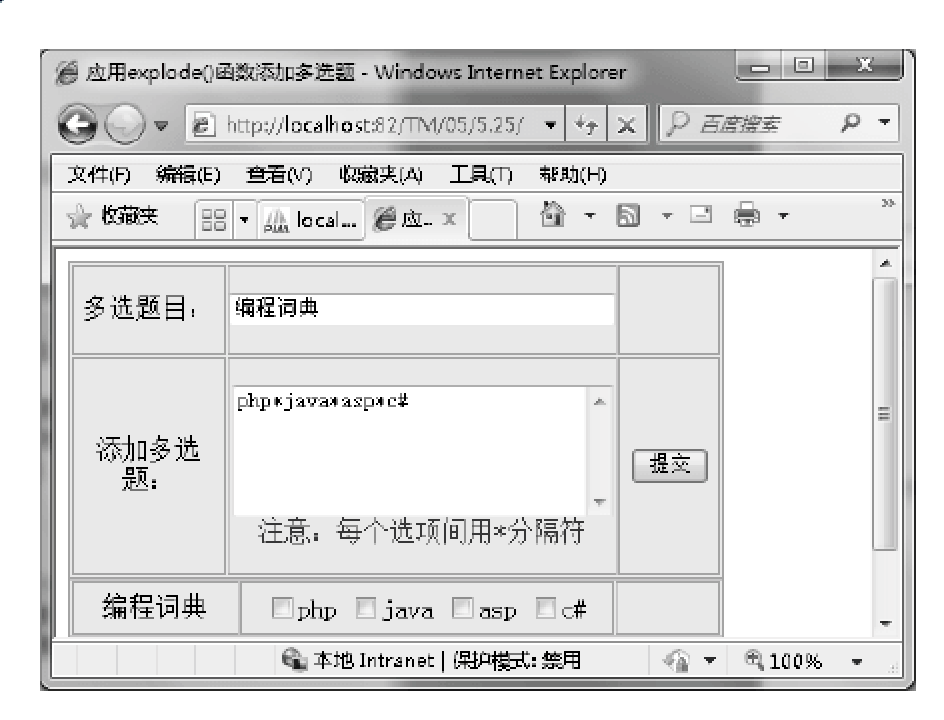


图 5.22　添加多选题功能

3．尝试开发一个页面，使用 sort()函数对指定的数组进行升序排序。**（答案位置：光盘\TM\Instance\05\5.26）**

第 6 章

日期和时间的管理

（视频讲解：43 分钟）

对日期和时间进行操作的程序无处不在，特别是在基于 Web 的应用程序中，如表单提交的时间、用户登录的时间、数据库中数据的更新和删除的时间等。想要记录这些操作执行的时刻，就需要通过日期和时间来完成。日期和时间在我们的生活中必不可少，同样，在互联网中，日期和时间也是非常重要的。本章就来讲解在 PHP 中是如何操作日期和时间的。

通过阅读本章内容，你可以：

- 掌握系统的时区设置
- 掌握如何获取本地时间戳
- 掌握如何获取当前日期和时间
- 掌握如何格式化日期和时间
- 掌握如何验证日期的有效性
- 掌握如何运用时间差

6.1　PHP 的时间概念

视频讲解：光盘\TM\Video\第 6 章\PHP 的时间概念.exe

在 PHP 语言中，日期、时间函数依赖于服务器的地区设置，而 PHP 默认设置的是标准的格林威治时间（即采用的是零时区）。如果没有对 PHP 的时区进行设置，并且用户的当地时间是北京时间，那么你通过 PHP 的时间函数获取的时间就将比当地的北京时间少 8 个小时。因此，要获取本地当前的时间必须更改 PHP 语言中的时区设置。更改 PHP 语言中的时区设置有两种方法：在 php.ini 文件中设置和通过 date_default_timezone_set 函数设置。

6.1.1　在 php.ini 文件中设置时区

视频讲解：光盘\TM\Video\第 6 章\在 php.ini 文件中设置时区.exe

在 php.ini 文件中设置时区，需要定位到[date]下的“;date.timezone =”选项，去掉前面的分号，并设置它的值为当地所在时区使用的时间。

例如，如果当地所在时区为东八区，就可以设置“date.timezone =”的值为：PRC（中华人民共和国）、Asia/Hong_Kong（香港）、Asia/Shanghai（上海）或者 Asia/Urumqi（乌鲁木齐）等，这些都是东八区的时间，如图 6.1 所示。

设置完成后，保存文件，重新启动 Apache 服务器。

6.1.2　通过 date_default_timezone_set 函数设置时区

由于 PHP 5 对 data()函数进行了重写，因此，目前的日期时间函数比系统时间少 8 个小时。PHP 语言默认设置的是标准的格林威治时间（即采用的是零时区），所以要获取本地当前的时间必须更改 PHP 语言中的时区设置。

在应用程序中，日期、时间函数之前使用 date_default_timezone_set()函数同样可以完成对时区的设置。date_default_timezone_set()函数的语法如下：

```
date_default_timezone_set(timezone);
```

参数 timezone 为 PHP 可识别的时区名称，如果时区名称 PHP 无法识别，则系统采用 UTC 时区。在 PHP 手册中提供了各时区名称列表，其中，设置我国北京时间可以使用的时区包括 PRC（中华人民共和国）、Asia/Chongqing（重庆）、Asia/Shanghai（上海）或者 Asia/Urumqi（乌鲁木齐），这几个时区名称是等效的。

设置完成后，date()函数便可正常使用，不会再出现时差问题。

例 6.1　通过 date()函数格式化输出当前时刻的 UTC 时间和北京时间。**（实例位置：光盘\TM\Instance\06\6.1）**

代码如下：

```
<?php
echo "UTC 时间：".date("Y-m-d H:i:s");                    //显示默认的 UTC 时间
date_default_timezone_set("PRC");                         //使用中华人民共和国的时区
echo "北京时间：".date("Y-m-d H:i:s");                    //输出北京时间
echo "当前时区：".date_default_timezone_get();            //获取当前时区
?>
```

运行结果如图 6.2 所示。



图 6.1　设置 PHP 的时区

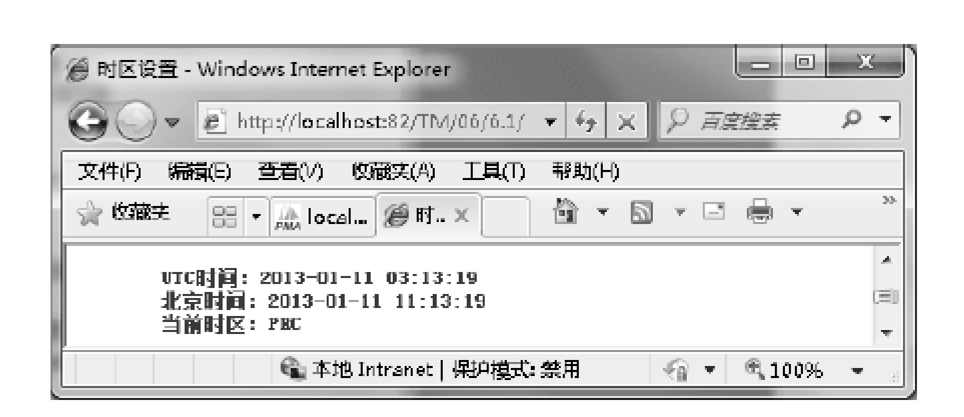


图 6.2　以不同的时区输出当前时间

技巧 如果服务器使用的是零时区，则不能对 php.ini 文件直接进行修改，只能通过 date_default_timezone_set()函数对时区进行设置。

6.2　时　间　戳

视频讲解：光盘\TM\Video\第 6 章\时间戳.exe

6.2.1　什么是时间戳

时间戳是文件属性中的创建、修改和访问时间。数字时间戳服务（Digita1 Time Stamp Service，DTS）是 Web 网站安全服务项目之一，能提供电子文件的日期和时间信息的安全保护。

时间戳是一个经加密后形成的凭证文档，它包括以下 3 个部分：

（1）需要添加时间戳的文件用 Hash 编码加密形成摘要。

（2）DTS 接收文件的日期和时间信息。

（3）对接收的 DTS 文件加密。

数字时间是由认证单位 DTS 来添加的，以 DTS 接收到文件的时间为依据。

时间戳的作用原理是通过其他加密法将时间的数值转换为加密的数值，时间变化后加密的数值也随之变化。

时间戳的优点是：变化的加密数值来防止数值被窃取后非法重复利用，也就起到了加密的作用。时间戳主要依赖于时间，在约定的一段时间内产生唯一的一个数值。

6.2.2　UNIX 时间戳

在 UNIX 系统中，日期与时间表示为自 1970 年 1 月 1 日零点起到当前时刻的秒数，这种时间称为 UNIX 时间戳，以 32 位二进制数表示。其中，1970 年 1 月 1 日零点称为 UNIX 世纪元。UNIX 时间戳提供了一种统一、简洁的时间表示方式，在不同的操作系统中均支持这种时间表示方式，同一时间在 UNIX 和 Windows 中均以相同的 UNIX 时间戳表示，所以不需要在不同的系统中进行转换。同时，UNIX 时间戳是一个时间差，与时区没有关系，无论当前 PHP 中使用的是何种时区，其 UNIX 时间戳是唯一的。

PHP 为 UNIX 时间戳的处理提供了各种函数。到目前的 PHP 版本为止，由于任何已知的 Windows 版本以及其他一些系统均不支持负的时间戳，因此在 Windows 中无法表示 1970 年 1 月 1 日之前的时间。目前 UNIX 时间戳是以 32 位二进制数表示的，32 位二进制数值范围为-2147483648～+2147483647，因此，目前 UNIX 时间戳可表示的最大时间为 2038 年 1 月 19 日 3 点 14 分 7 秒，该时刻时间戳为 2147483647，对于该

时刻之后的时间，需要扩展表示 UNIX 时间戳的二进制位数。

6.2.3　获取指定日期的时间戳

PHP 中应用 mktime()函数将一个时间转换成 UNIX 的时间戳值。

mktime()函数根据给出的参数返回 UNIX 时间戳。时间戳是一个长整数，包含了从 UNIX 纪元（1970 年 1 月 1 日）到给定时间的秒数。其参数可以从右向左省略，任何省略的参数都会被设置成本地日期和时间的当前值。即如果不设置任何参数，那么 mktime()函数获取的将是本地的当前日期和时间。

语法如下：

```
int mktime(int hour, int minute, int second, int month, int day, int year, int [is_dst] )
```

mktime()函数的参数说明如表 6.1 所示。

表 6.1　mktime()函数的参数说明

参　数	说　明
hour	小时数
minute	分钟数
second	秒数（一分钟之内）
month	月份数
day	天数
year	年份数，可以是两位或四位数字，0～69 对应于 2000～2069，70～100 对应于 1970～2000
is_dst	参数 is_dst 在夏令时可以被设为 1，如果不是则设为 0；如果不确定是否为夏令时则设为-1（默认值）

注意　有效的时间戳典型范围是格林威治时间 1901 年 12 月 13 日 20:45:54 到 2038 年 1 月 19 日 03:14:07（此范围符合 32 位有符号整数的最小值和最大值）。在 Windows 系统中此范围限制为从 1970 年 1 月 1 日到 2038 年 1 月 19 日。

例 6.2　应用 mktime()函数获取系统的当前日期时间，由于返回的是时间戳，所以还要通过 date()函数对其进行格式化后才能够输出日期和时间。**（实例位置：光盘\TM\Instance\06\6.2）**

代码如下：

```
<?php
    echo "时间戳：".mktime()."<br>";                    //返回当前的时间戳
    echo "日期：".date("Y-m-d",mktime())."<br>";        //使用 date()函数输出格式化后的日期
    echo "时间：".date("H:i:s",mktime());               //使用 date()函数输出格式化后的时间
?>
```

运行结果如图 6.3 所示。

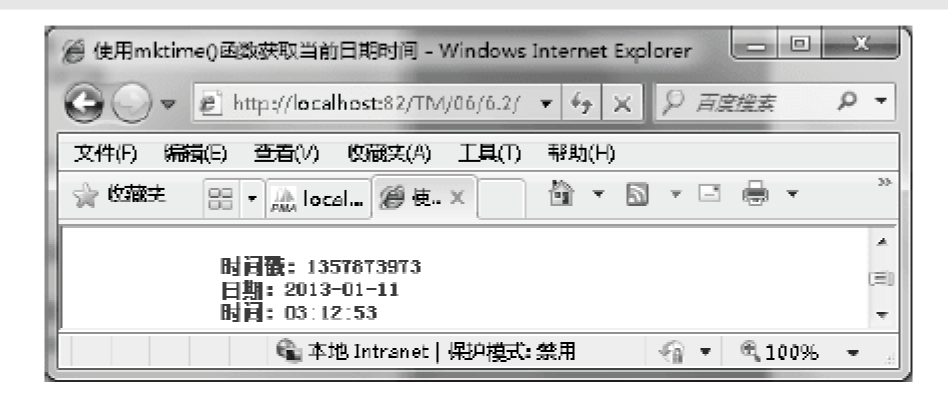


图 6.3　使用 mktime()函数获取当前日期时间

6.2.4　获取当前时间戳

PHP 通过 time()函数获取当前的 UNIX 时间戳，返回值为从 UNIX 纪元（格林威治时间 1970 年 1 月 1 日 00:00:00）到当前时间的秒数。

语法如下：

```
int time ( void )
```

该函数没有参数，返回值为 UNIX 时间戳的整数值。

例 6.3 应用 time()函数获取当前时间戳，并将时间戳格式化输出。（**实例位置：光盘\TM\Instance\06\6.3**）

代码如下：

```
<?php
    $nextWeek = time() + (7 * 24 * 60 * 60);             //7 days; 24 hours; 60 mins; 60secs
    echo time()."<br>";                                  //当前时间戳
    echo "Now：".date("Y-m-d")."<br>";                   //输出当前日期
    echo "Next Week：".date("Y-m-d",$nextWeek);          //输出变量 nextweek 的日期
?>
```

运行结果如图 6.4 所示。

6.2.5 将英文文本的日期时间描述解析为 UNIX 时间戳

PHP 中应用 strtotime()函数将任何英文文本的日期时间解析为 UNIX 时间戳，其值为相对于 now 参数给出的时间，如果没有提供此参数则用系统当前时间。

语法如下：

```
int strtotime ( string time [, int now] )
```

该函数有两个参数。如果参数 time 的格式是绝对时间，则 now 参数不起作用；如果参数 time 的格式是相对时间，其对应的时间就是参数 now 来提供的，当没有提供参数 now 时，对应的时间就为当前时间。如果解析失败，则返回 false。在 PHP 5.1.0 之前的版本中，本函数在失败时返回-1。

例 6.4 应用 strtotime()函数获取英文格式日期时间字符串的 UNIX 时间戳，并将部分时间输出。（**实例位置：光盘\TM\Instance\06\6.4**）

代码如下：

```
<?php
    echo strtotime ("now"), "\n";                                              //当前时间的时间戳
    echo "输出时间:".date("Y-m-d H:i:s",strtotime ("now")),"<br>";             //输出当前时间
    echo strtotime ("10 November 2012"), "\n";                                 //输出指定日期的时间戳
    echo "输出时间:".date("Y-m-d H:i:s",strtotime ("10 November 2012")),"<br>"; //输出指定日期的时间
    echo strtotime ("+3 day"), "\n";
    echo "输出时间:".date("Y-m-d",strtotime ("+3 day")),"<br>";
    echo "加一周：".strtotime ("+1 week")."<br>";
    echo "加一周两天三小时四分钟：".strtotime ("+1 week 2 days 3 hours 4 seconds")."<br>";
    echo "下周四：".strtotime ("next Thursday")."<br>";
    echo "上周一：".strtotime ("last Monday"), "\n";
?>
```

运行结果如图 6.5 所示。

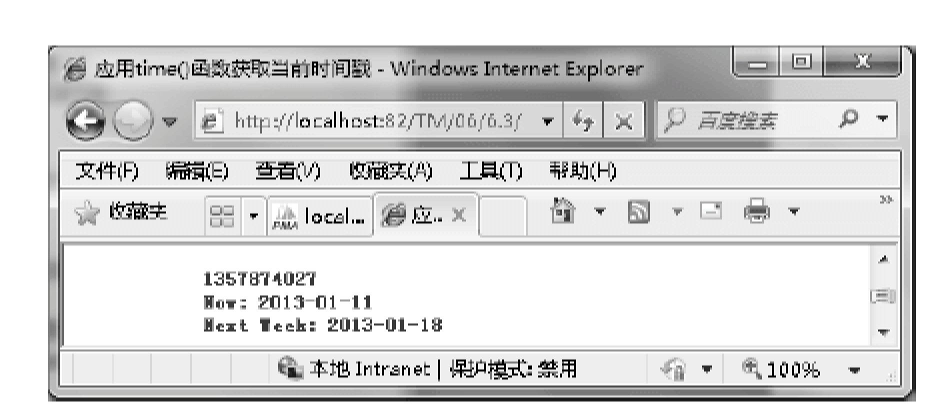


图 6.4 使用 time()函数获取当前时间戳

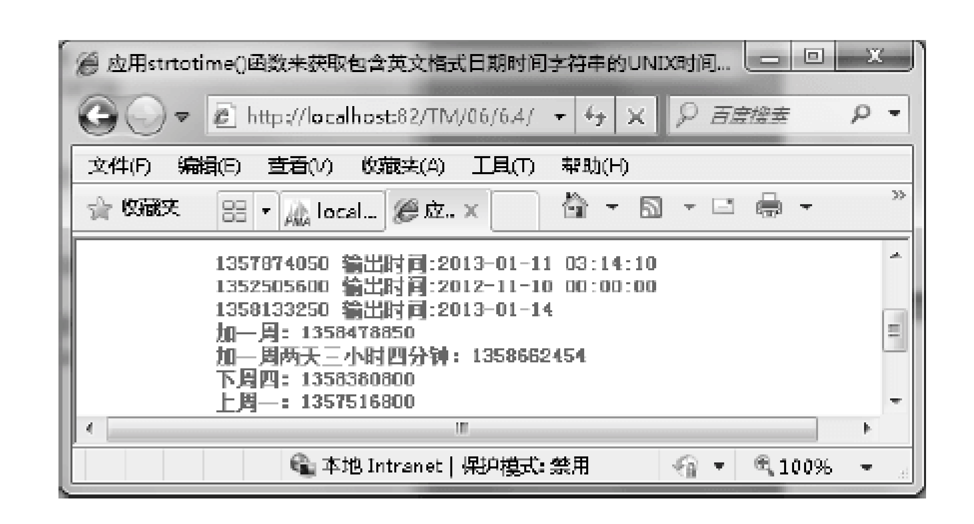
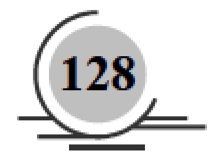


图 6.5 使用 strtotime()函数获取当前时间戳



6.3　PHP 日期和时间的处理

视频讲解：光盘\TM\Video\第 6 章\PHP 日期和时间的处理.exe

日期和时间的处理可以分为格式化日期和时间、获取日期和时间信息、获取本地化的日期和时间及检验日期和时间的有效性等。

6.3.1　格式化日期和时间

PHP 中通过 date()函数对本地日期和时间进行格式化。语法如下：

```
date(string format,int timestamp)
```

参数 format 指定日期和时间输出的格式。有关参数 format 指定的格式如表 6.2 所示。

表 6.2　参数 format 的格式化选项

参　数	说　明
a	小写的上午和下午值，返回值为 am 或 pm
A	大写的上午和下午值，返回值为 AM 或 PM
B	Swatch Internet 标准时间，返回值为 000～999
d	月份中的第几天，有前导零的两位数字，返回值为 01～31
D	星期中的第几天，文本格式，3 个字母，返回值为 Mon 到 Sun
F	月份，完整的文本格式，返回值为 January 到 December
G	小时，12 小时格式，没有前导零，返回值为 1～12
H	小时，24 小时格式，没有前导零，返回值为 0～23
i	有前导零的分钟数，返回值为 00～59
I	判断是否为夏令时，如果是夏令时返回值为 1，否则为 0
J	月份中的第几天，没有前导零，返回值为 1～31
l	星期数，完整的文本格式，返回值为 Sunday 到 Saturday
L	判断是否为闰年，如果是闰年返回值为 1，否则为 0
m	数字表示的月份，有前导零，返回值为 01～12
M	3 个字母缩写表示的月份，返回值为 Jan 到 Dec
n	数字表示的月份，没有前导零，返回值为 1～12
o	与格林威治时间相差的小时数，如 0200
r	RFC 822 格式的日期，如 Thu，21 Dec 2000 16:01:07 +0200
s	秒数，有前导零，返回值为 00～59
S	每月天数后面的英文后缀，两个字符，如 st、nd、rd 或者 th。可以和 j 一起使用
t	指定月份所应有的天数
T	本机所在的时区
U	从 UNIX 纪元（January 1 1970 00:00:00 GMT）开始至今的秒数
w	星期中的第几天，数字表示，返回值为 0～6
W	ISO-8601 格式年份中的第几周，每周从星期一开始
y	两位数字表示的年份，返回值如 88 或 08

续表

参　数	说　明
Y	4 位数字完整表示的年份，返回值如 1998、2008
z	年份中的第几天，返回值为 0～366
Z	时差偏移量的秒数。UTC 西边的时区偏移量总是负的，UTC 东边的时区偏移量总是正的，返回值为 -43200～43200

参数 timestamp 是可选的，用于指定时间戳，如果没有给出时间戳则使用本地当前时间 time()。

有关通过 date()函数获取系统当前时间的方法在前面的实例中已经应用过，这里不再赘述。

例 6.5　应用 date()函数对日期进行格式化，设置不同的参数，进而输出不同格式的日期。**（实例位置：光盘\TM\Instance\06\6.5）**

代码如下：

```
<?php
    echo date("Y-m-d")."<br>";                      //2012-11-10
    echo date("m.d.y")."<br>";                      //11.10.12
    echo date("j, n, Y")."<br>";                    //10, 11, 2012
    echo date("F j, Y, g:i a")."<br>";              //November 10, 2012, 5:12 am
    echo date("D M j G:i:s T Y")."<br>";            //Sat Nov 10 5:12:05 UTC 2012
    echo date('\I\t \i\s \t\h\e jS \d\a\y')."<br>"; //It is the 10th day
    echo date("H:i:s  这是当前时间")."<br>";         //05:12:05  这是当前时间
    echo date('h-i-s, j-m-y,这是我的一天')."<br>";    //05-12-05, 10-11-12,这是我的一天
?>
```

运行结果如图 6.6 所示。

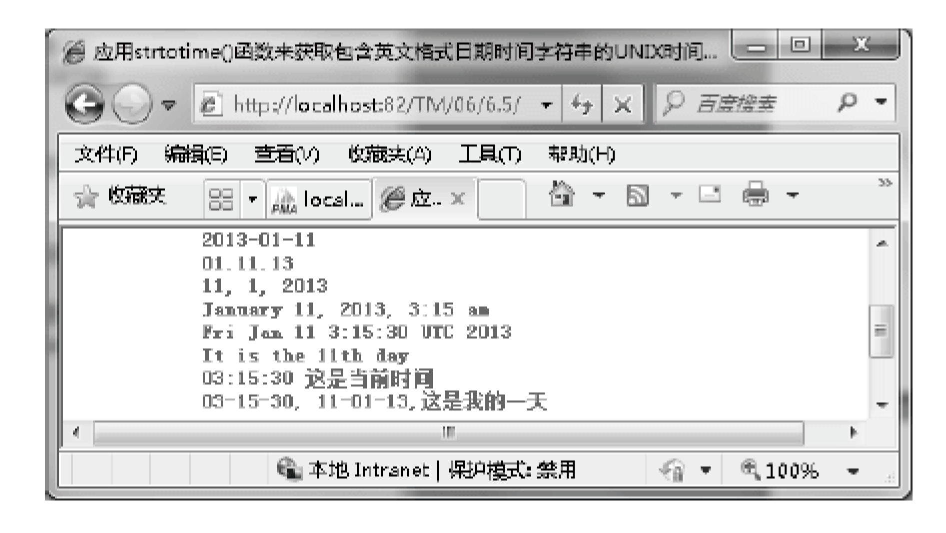


图 6.6　应用 date()函数对日期进行格式化

注意 在运行本章的实例时，也许有的读者得到的时间和系统时间并不相等，这不是程序的问题。因为在 PHP 语言中默认设置的是标准的格林威治时间，而不是北京时间。如果出现了时间不相符的情况，读者可参考 6.1.2 节中的内容。

6.3.2　获取日期和时间信息

PHP 中通过 getdate()函数获取日期和时间指定部分的相关信息。语法如下：

```
array getdate(int timestamp)
```

该函数返回数组形式的日期、时间信息，如果没有时间戳，则以当前时间为准。该函数返回的关联数组元素的说明如表 6.3 所示。

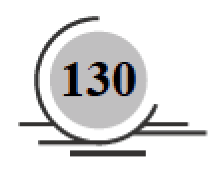


表 6.3 getdate()函数返回的关联数组元素说明

函 数	说 明
seconds	秒，返回值为 0～59
minutes	分钟，返回值为 0～59
hours	小时，返回值为 0～23
mday	月份中第几天，返回值为 1～31
wday	星期中第几天，返回值为 0（表示星期日）～6（表示星期六）
mon	月份数字，返回值为 1～12
year	4 位数字表示的完整年份，返回的值如 2000 或 2008
yday	一年中第几天，返回值为 0～365
weekday	星期几的完整文本表示，返回值为 Sunday 到 Saturday
month	月份的完整文本表示，返回值为 January 到 December
0	返回从 UNIX 纪元开始的秒数

例 6.6 应用 getdate()函数获取系统当前的日期信息，并输出返回值。（**实例位置：光盘\TM\Instance\06\6.6**）

代码如下：

```
<?php
    $arr = getdate();                                                   //使用 getdate()函数将当前信息保存
    echo $arr[year]."-".$arr[mon]."-".$arr[mday]." ";                  //返回当前的日期信息
    echo $arr[hours].":".$arr[minutes].":".$arr[seconds]." ".$arr[weekday];  //返回当前的时间信息
    echo "Today is the $arr[yday]th of year";                          //输出今天是一年中的第几天
?>
```

运行结果如图 6.7 所示。

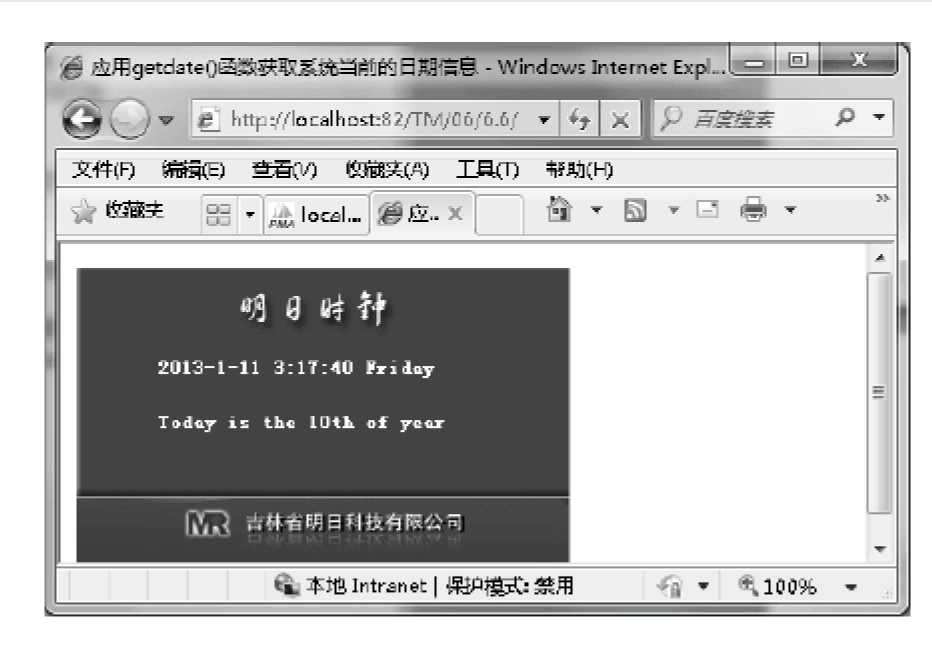


图 6.7 用 getdate()函数获取时间信息

6.3.3 获取本地化的日期和时间

不同的国家和地区，使用不同的时间、日期、货币的表示法，以及不同的字符集。例如，在大多数西方国家，都使用 Friday，但在以汉语为主的国家中，都使用星期五，虽然都是同一个含义，但表示的方式却不尽相同。这时，就需要设置本地化环境。

在 PHP 中应用 setlocale()函数设置本地化环境，应用 strftime()函数根据区域设置格式化本地日期和时间。

1. setlocale()函数

setlocale()函数可以改变 PHP 默认的本地化环境。语法如下：

```
string setlocale(string category, string locale)
```

参数 category 的选项如表 6.4 所示。参数 locale 如果为空，就会应用系统环境变量的 locate 或 LANG 的值；否则，就会应用 locale 参数所指定的本地化环境。如 en_US 为美国本地化环境，chs 则指简体中文，cht 为繁体中文。

说明 对于 Windows 平台的用户，可以登录 http://msdn.microsoft.com 来获取语言和国家（地区）的编码列表。如果是 UNIX/Linux 系统，则可以使用 locale–a 命令来确定所支持的本地化环境。

表 6.4　category 参数选项及说明

参　数	说　明
LC_ALL	包含了下面所有的设置本地化规则
LC_COLLATE	字符串比较
LC_CTYPE	字符串分类和转换，如转换大小写
LC_MONETARY	本地化环境的货币形式
LC_NUMERIC	本地化环境的数值形式
LC_TIME	本地化环境的时间形式

2. strftime()函数

strftime()函数根据区域设置来格式化输出日期和时间。语法如下：

```
string strftime(string format, int timestamp)
```

参数 format 指定日期和时间输出的格式。有关参数 format 识别的转换标记如表 6.5 所示。参数 timestamp 指定时间戳，如果没有指定时间戳则应用本地时间。

表 6.5　参数 format 识别的转换标记

参　数	说　明
%a	星期的简写
%A	星期的全称
%b	月份的简写
%B	月份的全称
%c	当前区域首选的日期时间表达
%C	世纪值（年份除以 100 后取整，范围为 00～99）
%d	月份中的第几天，十进制数字（范围为 01～31）
%D	和%m/%d/%y 相同
%e	月份中的第几天，十进制数字，一位的数字前会加上一个空格（范围为' 1'～'31'）
%g	和%G 相同，但是没有世纪值
%G	4 位数的年份，符合 ISO 星期数（参见%V）。与%V 的格式和值相同，但如果 ISO 星期数属于前一年或者后一年，则使用那一年
%h	和%b 相同
%H	24 小时制的十进制小时数（范围为 00～23）
%I	12 小时制的十进制小时数（范围为 00～12）
%j	年份中的第几天，十进制数（范围为 001～366）
%m	十进制月份（范围为 01～12）
%M	十进制分钟数
%n	换行符
%p	根据给定的时间值为 am 或 pm，或者当前区域设置中的相应字符串
%r	用 a.m 和 p.m 符号表示的时间
%R	24 小时符号的时间
$S	十进制秒数
%t	制表符
%T	当前时间，和%H:%M:%S 相同
%u	星期几的十进制数表达[1,7]，1 表示星期一

续表

参　数	说　明
%U	本年的第几周，从第一周的第一个星期天作为第一天开始
%V	本年第几周的 ISO 8601:1988 格式，范围为 01～53，第一周是本年第一个至少还有 4 天的星期，星期一作为每周的第一天（用%G 或者%g 作为指定时间戳相应周数的年份组成）
%W	本年的第几周数，从第一周的第一个星期一作为第一天的开始
%w	星期中的第几天，星期天为 0
%x	当前区域首选的时间表示法，不包括时间
%X	当前区域首选的时间表示法，不包括日期
%y	没有世纪数的十进制年份（范围为 00～99）
%Y	包括世纪数的十进制年份
%Z	时区名或缩写
%%	文字上的%字符

例 6.7　应用 setlocale()函数进行区域化设置，然后通过 strftime()函数对本地时间进行格式化输出。**（实例位置：光盘\TM\Instance\06\6.7）**

代码如下：

```
<?php
	setlocale(LC_ALL,"en_US");
	echo "美国格式：".strftime("Today is %A");
	echo "<p>";
	setlocale(LC_ALL,"chs");
	echo "中文简体格式：".strftime("今天是%A");
	echo "<p>";
	setlocale(LC_ALL,"cht");
	echo "<p>";
	echo "繁体中文格式：".strftime("今天是%A");
?>
```

运行结果如图 6.8 所示。

说明　因为本页面中的编码格式为 GB2312，所以最后繁体中文显示的日期为乱码，如果将编码格式改为 big5，繁体中文星期将会正确显示出来，但其他文字则变为乱码。读者可以选择“查看”/“编码”命令，在弹出的菜单中选择“繁体中文(big5)”命令来查看效果。

6.3.4　检验日期和时间的有效性

一年有 12 个月，一个月有 31 天（或 30 天、28 天、29 天），一星期有 7 天……这些常识都是人尽皆知的事。但计算机并不能自己分辨数据的对与错，只能依靠开发者提供的功能去执行或检查。PHP 中通过 checkdate()函数检验日期和时间的有效性。语法如下：

```
bool checkdate(int month,int day,int year)
```

其中，month 的有效值为 1～12。day 的有效值为当月的最大天数，如 1 月为 31 天、2 月为 29 天（闰年）。year 的有效值为 1～32767。

例 6.8　应用 checkdate()函数验证数据录入系统中输入的日期是否合理，如果合理则返回“数据录入成功”，否则返回“您输入的日期不合法！！”。**（实例位置：光盘\TM\Instance\06\6.8）**

操作步骤如下所示。

（1）创建 index.php 页面，添加 form 表单，设置表单元素，用于提交商品的录入信息，将表单中的数据提交到 index_ok.php 文件中。

（2）创建 index_ok.php 文件，通过$_POST 全局变量获取表单中提交的数据，应用 checkdate()函数对表单中提交的日期进行判断。代码如下：

```
<?php
    if($_POST[Submit]==true and $_POST[number]==true and $_POST[name]==true and $_POST[day]
==true ){                                               //判断表单中提交的数据是否为空
        if(checkdate($month,$day,$year)==true){         //验证表单中提交的日期是否合理
            echo "数据录入成功!!";
        }else{
            echo "<script> alert('您输入的日期不合法!!'); history.back();</script>";
        }
    }else{
        echo "<script> alert('请详细填写录入数据!!'); history.back();</script>";
    }
?>
```

运行结果如图 6.9 所示。

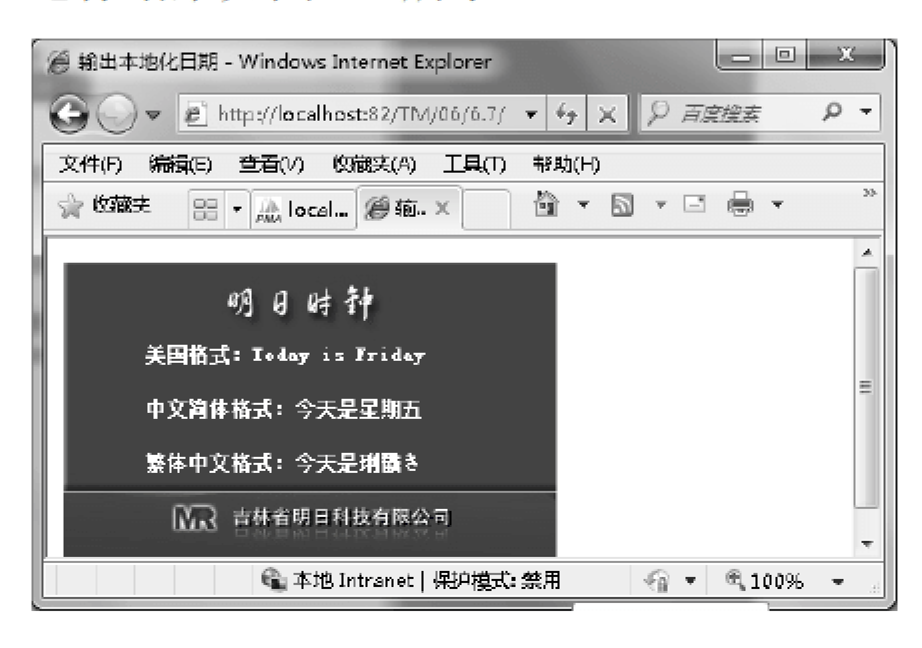


图 6.8　本地化日期

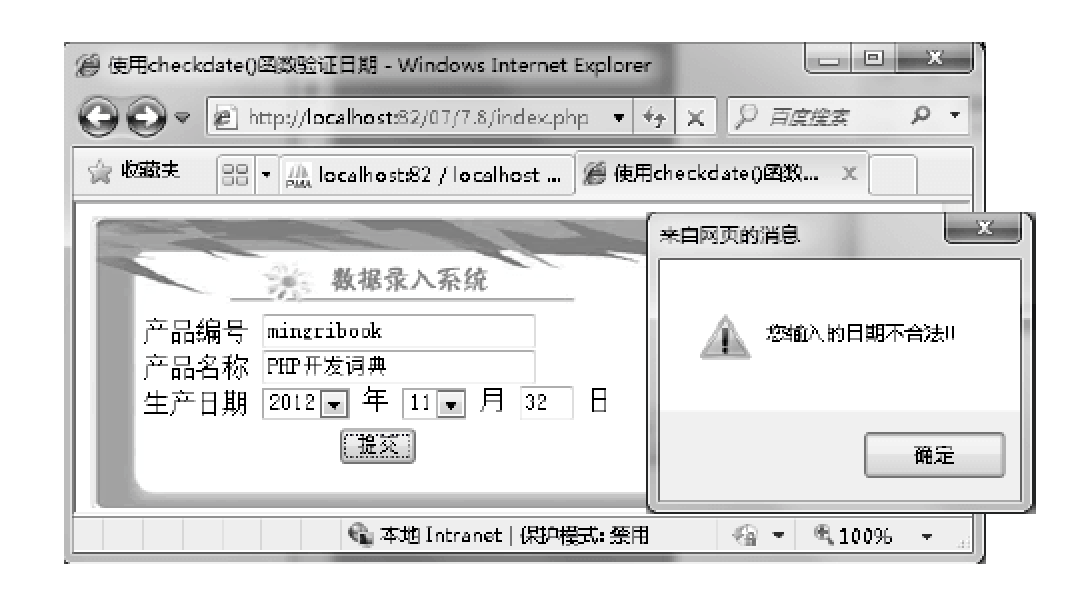
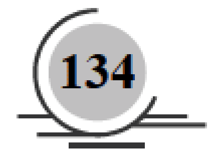


图 6.9　使用 checkdate()函数验证日期

6.4　实　　战



视频讲解：光盘\TM\Video\第 6 章\实战.exe

6.4.1　实现倒计时的功能

例 6.9　应用 strtotime()函数精确计算两个时间的差值，实现一个倒计时的功能。（**实例位置：光盘\TM\Instance\06\6.9**）

实例代码如下：

```
<?php
    $time1 = strtotime(date( "Y-m-d H:i:s"));
    $time2 = strtotime("2012-11-11 00:00:00");          //2012 年光棍节
    $time3 = strtotime("2013-01-01 ");                  //2013 年元旦
    $sub1 = ceil(($time2 - $time1) / 3600);             //60 * 60
    $sub2 = ceil(($time3 - $time1) / 86400);            //60 * 60 * 24
    echo "距离 2012 年光棍节还有<font color=red> $sub1 </font>小时!!!" ;
```

```
    echo "<p>";
    echo "距离 2013 年元旦还有<font color=red>$sub2 </font>天!!!";
?>
```

说明 ceil()函数的格式为 float ceil(float value)，该函数为取整函数，返回不小于参数 value 值的最小整数。如果有小数部分，则进一位。这里要注意，该函数的返回类型为 float 型，而不是整型。

运行结果如图 6.10 所示。



图 6.10　实现倒计时的功能

6.4.2　计算在线考试用时和剩余时间

例 6.10　在线考试系统中，多数都是在规定的时间内完成考试，如果超过规定时间，或者自动提交答案，或者自动放弃考试。本实例将要讲解的就是如何计算考试的用时和剩余时间。（**实例位置：光盘\TM\Instance\06\6.10**）

具体操作步骤如下。

（1）创建 index.php 页面。首先介绍考试规则；然后添加 form 表单，设置"开始考试"按钮，提交到 exam.php 页面；最后应用 mktime()函数获取当前日期的时间戳，并把时间戳的值赋给 session 变量。代码如下：

```
<?php
    session_start();                              //初始化 session 变量
?>
<form name="form2" method="post" action="exam.php">
    <input type="submit" name="Submit" value="开始考试" />
<?php
    $dates=mktime();                              //获取当前时间的时间戳
    session_register("dates")==$dates;            //将时间戳的值赋给 session 变量
?>
</form>
```

注意 在线考试时间的计算起点就是从单击"开始考试"按钮开始。

（2）创建 exam.php 文件。首先通过<script>标记中的 src 属性，调用屏蔽键盘事件和计算考试时间的 JS 文件；然后在<body>标记中，通过 onkeydown()事件调用 JavaScript 自定义函数，实现对键盘中事件的屏蔽；接着，通过<div>标记输出考试用时和剩余时间；最后，添加 form 表单，输出考试试题，设置提交按钮。代码如下：

```
<?php session_start();    ?>
<!--应用<script>标记中的 src 属性，调用屏蔽键盘事件和计算考试时间的 JS 文件-->
<script language=javascript src="js/screen_event.js"></script>
<script type="text/javascript" src="js/calculation_time.js"></script>
<body onkeydown="keydown()">
<table width="750" bordercolor="#FFFFFF" bgcolor="#666666">
    <tr>
        <td width="165" height=23 ><span class="STYLE1">考试时间：</span></td>
        <td width="42" ><span class="STYLE1"><font color="#FF0000">20</font>分钟</span></td>
```

```
            <td width="58" ><span class="STYLE1">计  时</span></td>
            <td width="193" nowrap bgcolor="#EEEEEE"><div class="STYLE1" id="show_time"></div></td>
            <td width="77" ><span class="STYLE1">剩余时间：</span></td>
            <td width="182" ><div class="STYLE1" id="sparetime"></div></td>
      </tr>
</table>
<!--省略了添加的 form 表单中的内容  -->
</body>
</html>
```

（3）创建 calculation_time.js 文件，通过 Ajax 无刷新技术计算和输出考试用时和剩余时间。关键代码如下：

```
var xmlHttp = false;                                      //创建一个布尔型变量，用来检测是否为合法的 IE 实例
try {                                                     //检测是否使用的是 IE
      xmlHttp = new ActiveXObject("Msxml2.XMLHTTP"); //如果 JavaScript 的版本大于 5
} catch (e) {                                             //如果不是，则使用老版本的 ActiveX 对象
      try {                                               //如果使用的是 IE 浏览器
            xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
      } catch (e2) {}
}
/* ******************如果使用的是非 IE 浏览器，则创建一个该对象的 JavaScript 实例******************** */
if (!xmlHttp && typeof XMLHttpRequest != "undefined") {
      try{
            xmlHttp = new XMLHttpRequest();
      }catch(e3){ xmlHttp = false;}
}

timer = window.setInterval("ShowTime()",1000);            //每隔一秒钟调用一次 ShowTime()函数
//定义 ShowTime()函数以通过 xmlHttpRequest 对象读取 ShowTime.php 文件中的数据
function ShowTime(){
      xmlHttp.open("post","showtime.php", true);          //以 POST 方法发送一个新请求
      xmlHttp.onreadystatechange = function(){
      if(xmlHttp.readyState == 4){                        //如果服务器响应发出请求，则执行以下操作
            tet = xmlHttp.responseText;                   //获取返回的响应信息
            /* *********************将获取到的信息赋予指定的 DIV 标记******************** */
            document.getElementById("show_time").innerHTML = tet;
      }
}
xmlHttp.send(null);                                       //发送请求
}


time = window.setInterval("sparetime()",1000);            //每隔一秒钟调用一次 sparetime()函数
/* ********定义 sparetime()函数以通过 xmlHttpRequest 对象读取 sparetime.php 文件中的数据************* */
function sparetime(){
      xmlHttp.open("post","sparetime.php", true);         //以 POST 方法发送一个新请求
      xmlHttp.onreadystatechange = function(){
            if(xmlHttp.readyState == 4){                  //如果服务器响应发出请求，则执行以下操作
                  tet = xmlHttp.responseText;             //获取返回的响应信息
                  /* **********************将获取到的信息赋予指定的 DIV 标记********************** */
                  document.getElementById("sparetime").innerHTML = tet;
                  if(tet=="00:00"){                       //当剩余时间为 00:00 时
                        form1.submit();                   //提交 form1 表单中的数据
                  }
```

```
            }
        }
        xmlHttp.send(null);                              //发送请求
}
```

（4）在 calculation_time.js 文件中，自定义函数 ShowTime()调用 showtime.php 文件，获取考试用时；自定义函数 sparetime()调用 sparetime.php 文件，获取考试的剩余时间。

showtime.php 文件中计算的是考试用时。代码如下：

```
<?php
        session_start();                                 //初始化 session 变量
        $dates=$_SESSION[dates];                         //获取 session 变量中存储的考试开始时间
       $dates2=mktime();                                 //获取当前时间的时间戳
       $dates3=$dates2-$dates;                           //用当前时间的时间戳减去考试开始时的时间戳
        echo date("i:s",$dates3);                        //输出考试当前用时
?>
```

sparetime.php 文件中计算的是考试剩余时间。代码如下：

```
<?php
        session_start();                                 //初始化 session 变量
        $dates=$_SESSION[dates];                         //获取 session 中存储的考试开始的时间戳
       $dates1=$dates+20*60;                             //计算出考试规定的用时，时间为 20 分钟
        $dates2=mktime();                                //获取当前时间的时间戳
       $dates3=$dates1-$dates2;                          //用考试规定的时间减去当前时间的时间戳
        echo date("i:s",$dates3);                        //输出考试的剩余时间

?>
```

（5）创建 screen_event.js 文件，编写自定义函数，屏蔽键盘和鼠标事件。

（6）创建 exam_result.php 文件，输出考试的结果。

在线考试的主页面运行结果如图 6.11 所示。



图 6.11　计算在线考试用时和剩余时间

6.4.3 网页闹钟

例 6.11 所谓网页闹钟也就是一个日志提醒功能，通过判断系统中当前的时间与指定的某个时间或者时间段是否相同，如果相同系统就给出一个对应的提示信息，提示用户该做什么（如每天工作记录的上传、每月经验技巧的提交或者员工生日的提醒等），都可以通过日志的形式进行提醒。在本实例中，通过当前时间戳与 11 月 10 日时间戳进行比较，如果相同则给出提示信息。**（实例位置：光盘\TM\Instance\06\6.11）**

代码如下：

```
<?php
	$time1 = strtotime(date("Y-m-d"));				//当前的系统时间，获取月和天的时间戳
	$time2 = strtotime(date("Y")."-11-09");			//设置时间戳
	if($time1==$time2){						//判断两个时间戳是否相同
		echo "<script>alert('勿忘国耻！');window.location.href='index.php';</script>";		//给出提示信息
	}else{
		echo "今天不是一个特殊的日子！";
	}
?>
```

运行结果如图 6.12 所示。

6.4.4 检验日期和时间的有效性

例 6.12 一年 12 个月、一个月 31 天（或 30 天，2 月 28 天，闰年为 29 天），一星期 7 天……这些都是基本常识。但计算机并不能自己分辨数据的对与错，只是依靠开发者提供的功能去执行或检查。在 PHP 中通过 checkdate()函数检验日期和时间的有效性。**（实例位置：光盘\TM\Instance\06\6.12）**

代码如下：

```
<?php
	header("Content-Type:text/html;charset=utf-8");
	$year = 2012;						//2012 年
	$month =6;							//6 月份
	$one_day =30;						//30 天
	if(checkdate($month,$one_day,$year)){
		echo "<font style='color:black;font-family:文鼎霹雳体;font-size:25'>2012 年 6 月份是 30 天</font>";
	}else{
		echo "<font style='color:black;font-family:文鼎霹雳体;font-size:18'>2012 年 6 月份是 31 天</font>";
	}
?>
```

运行结果如图 6.13 所示。

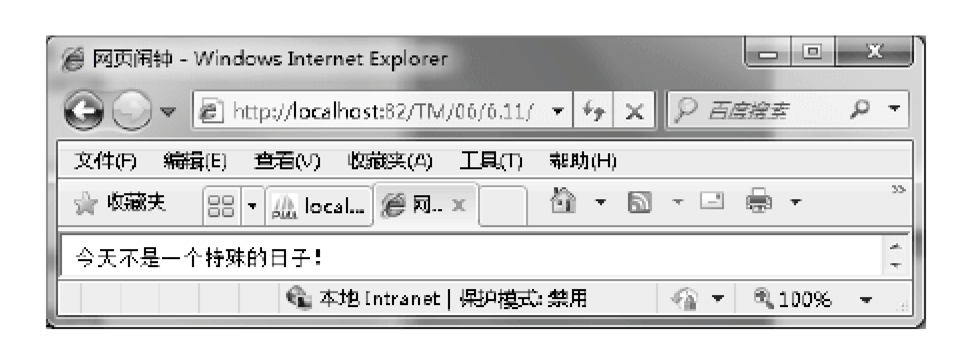


图 6.12 计算程序运行时间

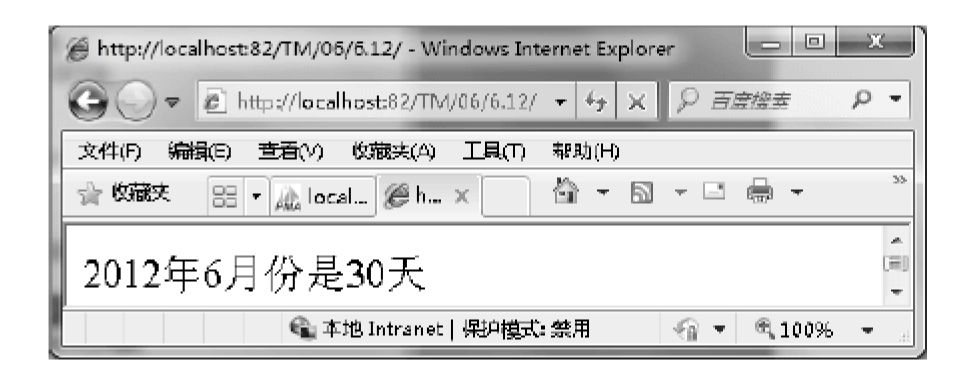
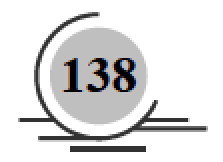


图 6.13 检验日期和时间的有效性



6.4.5　获取指定时间的 UNIX 时间戳

例 6.13　获取指定时间的时间戳需要使用 strtotime()函数，此函数接受任何一个英文文本的日期时间并将其转换为时间戳，本实例以 10 September 2010 为例获取此时刻的时间戳。**（实例位置：光盘\TM\Instance\06\6.13）**

实现原理非常简单，在 strtotime()函数中添加指定时间参数即可。代码如下：

```
<?php
    header("Content-Type:text/html;charset=utf-8");                                   //设置编码集
    $str="<font style='color:red;font-size:20px'>".strtotime("20 December 2012")."</font>";   //指定获取时间戳
    echo "20 December 2012 的时间戳为 ";
    echo $str;
?>
```

运行结果如图 6.14 所示。

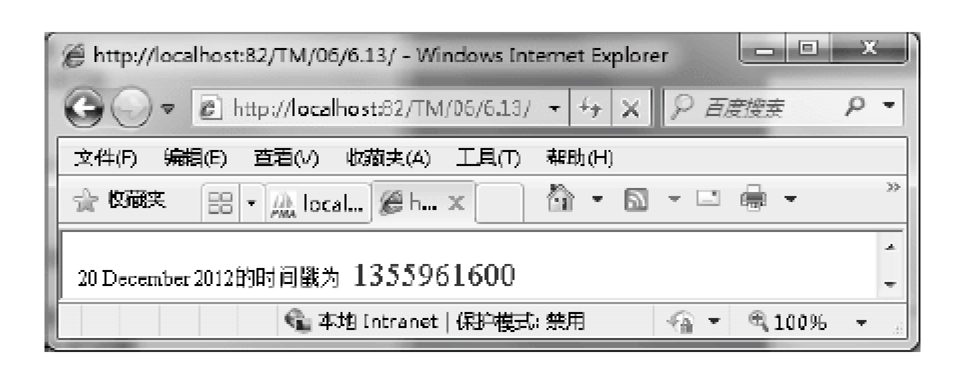


图 6.14　获取指定时间的时间戳

6.5　小　　结

本章讲解了 PHP 中常用的日期和时间处理函数，并且通过实例讲解了这些函数的典型应用。首先介绍时区的划分和系统时区的设置；然后介绍了 UNIX 时间戳，包括什么是时间戳、如何获取时间戳，以及如何将英文文本格式的日期时间解析为时间戳；接着，讲解了 PHP 中日期和时间的处理方法，包括日期和时间的格式化、获取和验证有效性等；最后，通过实战和实战练习讲解了 PHP 中日期和时间函数的应用。希望通过本章的学习，读者可以掌握 PHP 中常用的日期和时间函数，熟悉常用功能的实现方法。

6.6　学习成果检验

1. 获取指定任意一天的时间。格式为 YYY-MM-DD HH:MM:SS。**（答案位置：光盘\TM\Instance\06\6.14）**
2. 将时间戳作为随机数的种子，生成验证码。**（答案位置：光盘\TM\Instance\06\6.15）**
3. 如何计算当前时间距离下班时间还有多少时间？**（答案位置：光盘\TM\Instance\06\6.16）**

第 7 章

程序调试与异常处理

（视频讲解：72 分钟）

即使是具有多年开发经验的程序开发人员，在开发项目时也难免会发生代码编写错误，发生错误其实并不可怕，只要认真分析、努力思考就一定能够将问题解决。调试既是一种技术，也是一种艺术。随着经验的不断积累，我们就会更熟练地发现并且纠正错误，这其中的关键就是勇敢地去面对错误，不要气馁。本章将介绍一些调试 PHP 和 MySQL 代码的实用技巧，通过本章的学习定能使初学者受益匪浅、能力有所提高。

通过阅读本章内容，你可以：

- **了解程序调试的基本流程**
- **了解 PHP 中常见错误类型**
- **了解 PHP 中常见语法错误**
- **掌握 PHP 的基本调试策略**
- **掌握如何处理 MySQL 数据库中文的乱码问题**

7.1　程序基本调试流程

即使是一名具有多年开发经验的编程人员，在编写代码时也不可能一次通过，往往要经过多次调试处理才能实现某一功能。具备基本的调试能力和策略，不仅可以拓展开发人员的编程思路，更能有效缩短项目的整体开发周期，是开发人员所必备的素质。在实际应用中，程序的基本调试流程如图 7.1 所示。

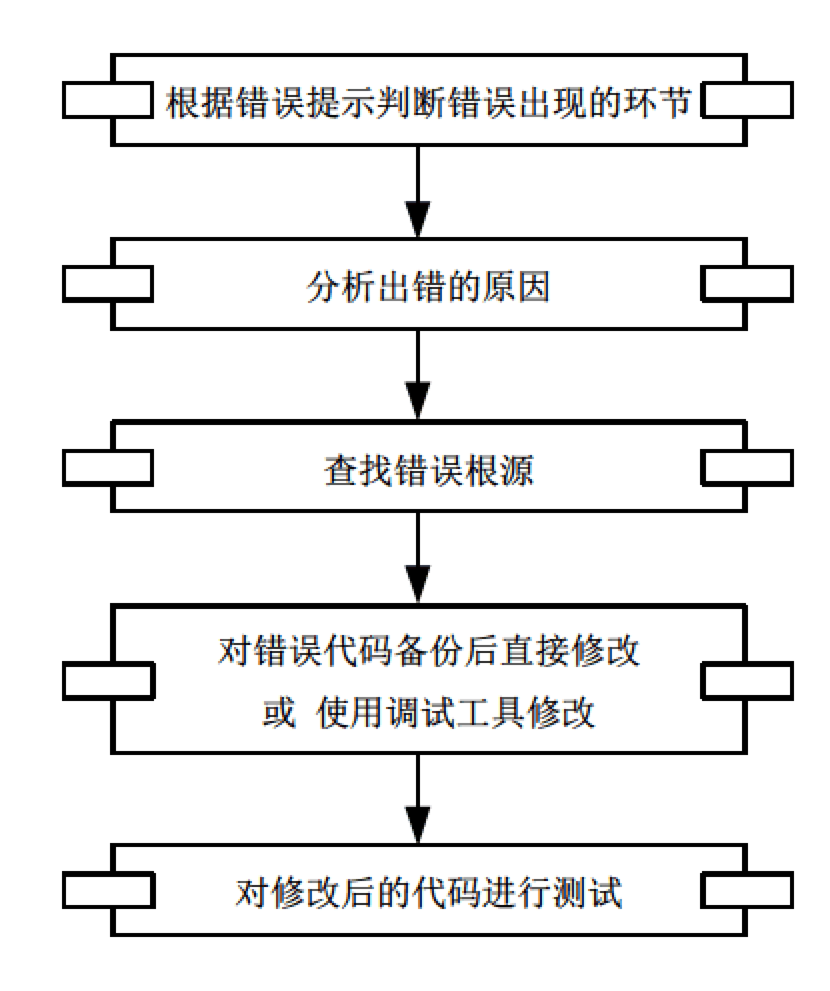


图 7.1　程序基本调试流程

从上述调试流程中可以判断，错误调试的过程是一个往复的过程，也就是说，在对程序中的错误进行调试时，可能要进行多次修改才能成功。在实际开发中，开发人员可以根据自己的经验选择适合自己的错误调试方法。下面具体介绍错误调试的一般流程。

（1）根据错误提示判断错误原因：程序错误一般分为开发时错误和运行时错误。开发时错误一般是由于开发人员的关键字书写错误造成的，现在很多开发工具都支持开发时错误提示，例如，在开发 PHP 程序时，可以使用 Zend Studio For Eclipse 这个开发工具捕获开发时错误。运行时错误是由于开发人员代码逻辑编写错误等原因造成的，一般可由编译工具在代码编译阶段提示错误，开发人员根据错误提示就可以定位到错误代码的大概位置。

（2）分析出错原因：定位到出错代码的大概位置后，根据错误提示分析错误原因，如关键字拼写错误、编码逻辑错误等。

（3）查找错误根源：分析出错的基本原因后，还需要进一步分析错误根源，不要认为只将错误改正就可以了，读者在学习编写程序时要有知其然、并知其所以然的精神，这样才能不断积累提高。

（4）备份代码：找到出错原因后，不应该立即对代码进行更改，首先需要备份出错代码，防止在更改过程中造成更大错误。

（5）修改可疑代码：完成代码备份后，开发人员可以对错误代码进行修改，之后再运行调试，也就是说，如果代码修改后再一次发生错误，那么再回到调试流程的第（1）步。

上面介绍的程序调试流程是最基本的调试过程，读者不必循规蹈矩，可以在开发中根据实际情况总结出适合自己的调试方法和流程。

7.2　PHP 中的错误类型

在实际项目开发中，可能会遇到各种错误，不过只要开发人员在平时学习和工作中多积累、多总结，就可以在开发过程中游刃有余，轻松地解决程序中的问题。在项目开发中常见的错误类型包括语法错误、语义错误、逻辑错误和注释错误等。

7.2.1　语法错误

视频讲解：光盘\TM\Video\第 7 章\语法错误.exe

每一种计算机语言都有自身的语法格式，开发人员必须按照这种语法规则书写代码，否则会发生语法

错误。PHP 语言自身有其独特的语法规则，如变量名前要加“$”、每行要以“;”结束、关键字大小写敏感等。不过，PHP 自身是一种弱声明语言，也就是说 PHP 中的变量不需声明即可使用，这就为开发人员带来了很大的方便。PHP 中常见的语法错误有如下几种形式。

1．缺少结束符引起的错误

编写 PHP 代码时，要求每一行以“;”结束，如果代码编写人员因疏忽未写结束符“;”，在运行或调试程序时就会发生错误。

例 7.1　使用 for 循环输出 1～20 之间的数字，并设置缺少结束符错误。**（实例位置：光盘\TM\Instance\07\7.1）**

```
<?php
    for($i=1; $i<21; $i++)
    {
    echo $i                                    //没写结束符
    }
?>
```

运行上述代码，将在页面中输出如图 7.2 所示的错误提示。

分析页面中的错误，从语意中可以判断程序的第 5 行应该以“;”结束，而不应该以“}”结束，也就是说，程序将“}”当作第 5 行的结束符，而非 for 循环体的结束标志。

2．缺少单引号或双引号引起的语法错误

和其他语言不同，PHP 中无论使用单引号还是双引号，所引部分都当作字符串常量，区别是使用单引号的效率较双引号高，而双引号字符串中可以包含变量并能进行区分。在编写代码时，开发人员可能由于书写错误，使用单引号或者双引号时少书写一个，或字符串两侧使用的引号不一致而导致语法错误。

例 7.2　使用 echo 语句输出一个字符串，并得到引号不一致错误。**（实例位置：光盘\TM\Instance\07\7.2）**

```
<?php
    echo "mingri';        //引号不一致
?>
```

运行上述实例，将在页面中输出如图 7.3 所示的错误提示。

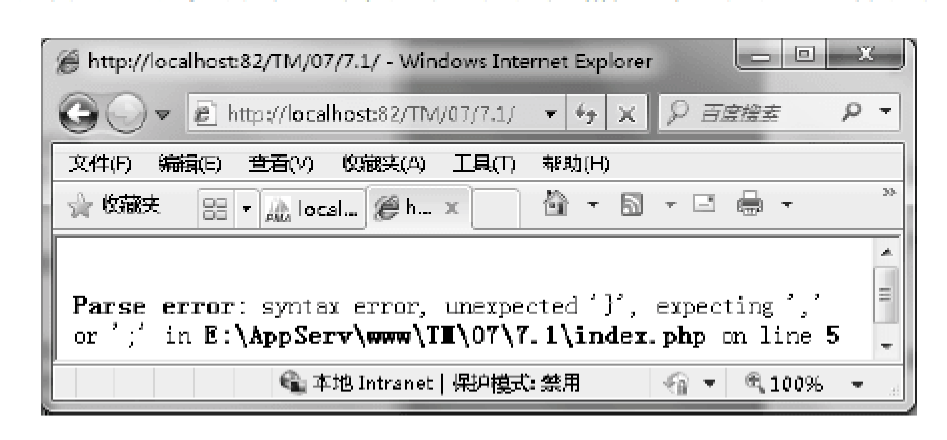


图 7.2　缺少结束符错误

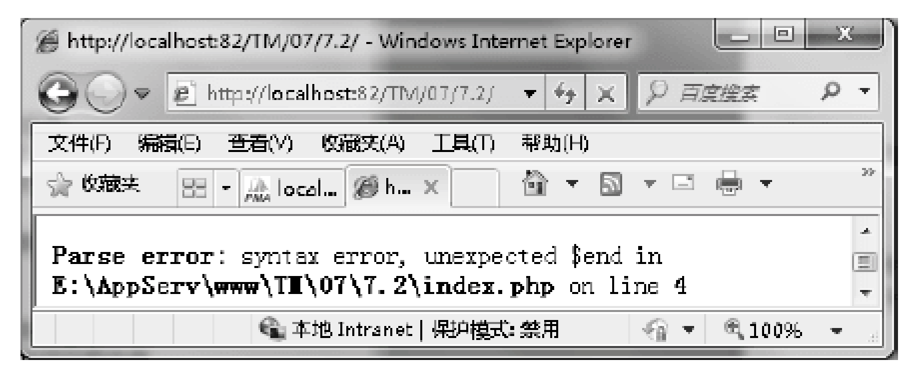


图 7.3　引号不一致错误

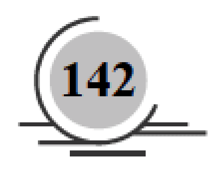
从错误提示中可以判断，导致错误的原因是没有结束标志，这说明在程序编译时将字符串后的单引号当作字符串的一部分，而非字符串的结束标志。

3．缺少括号引起的语法错误

与 C/C++语法类似，PHP 中诸如 for 循环、while 循环以及包含多条语句的 if 代码块都需要使用大括号。如果代码行数较多，很可能造成大括号遗漏。

例 7.3　使用双层 for 循环输出两个数的乘积，并得到内层循环缺少结束大括号错误。**（实例位置：光盘\TM\Instance\07\7.3）**

```
<?php
for($i=0; $i<10; $i++)
{
    for($j=0; $j<10; $j++)
```



```
    {
        echo $i*$j."<br>";
                                                    //遗漏循环体结束大括号
}
?>
```

运行上述实例，将在页面中输出如图 7.4 所示的错误提示。从错误提示中可以得知内层嵌套无结束标志，这就是未写内层 for 循环的结束大括号所导致的。

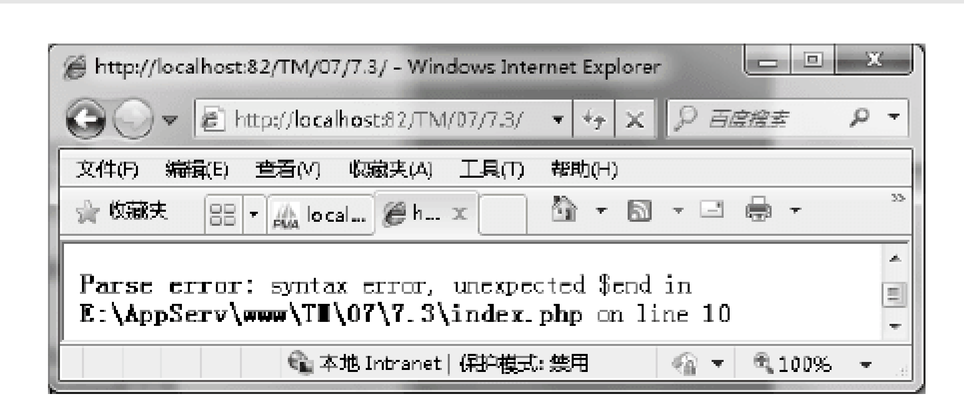


图 7.4　缺少 for 循环结束大括号错误提示

4．缺少变量标识符$引起的语法错误

在 PHP 中，设置变量时需要使用美元符号“$”，如果不添加美元符号就会引起解析错误。

例 7.4　下面通过实例讲解在使用变量时不添加美元符号会产生什么错误。（**实例位置：光盘\TM\Instance\07\7.4**）

```
<?php
  for ( $i=1 ; $i<11; i++){                         //缺少一个变量的美元符号
        echo $i."<br>";
  }
?>
```

运行程序时，输出如下错误提示信息：

Parse error: syntax error, unexpected T_INC, expecting ')' in **E:\AppServ\www\07\7.4\index.php** on line **2**

在该语句中，“i++”应该修改为“$i++”，如果是前面的“$i<11”没有使用美元符号，那么该程序将进入无限循环的状态直到服务器终止执行。

语法的错误是最基本的错误，只要在编写程序时认真一些，就会减少很多麻烦。这里介绍的只是语法错误中常见的几种类型，还有很多类似的错误，要避免错误的出现就要靠广大程序设计人员平时编写代码时多加注意，尽量书写准确的代码。

7.2.2　语义错误

语义错误是在语法正确的前提下导致的错误。例如，应用 PHP 连接符实现两个字符串的连接。代码如下：

```
<?php
    $str="PHP 开发实战宝典";
    $url="www.mingribook.com";
    echo $str + $url;                   //错误地使用了字符串连接符号，正确的连接符是“.”
?>
```

由于 PHP 中的字符串连接符是“.”，而不是“+”。上面的代码错误地使用了“+”作为字符串的连接符。但是由于 PHP 能够隐式转换变量类型，上面的代码并不会导致编译器出错，只是不会输出正确的结果。

7.2.3　逻辑错误

逻辑错误对于 PHP 编译器来说并不算错误，但是由于代码中存在的逻辑问题，导致运行结果没有得到期望的结果。逻辑错误在语法上是不存在错误的，但从程序的功能上看是缺陷，它是最难调试和发现的，因为它们不会抛出任何错误信息，唯一能看到的就是程序的功能（或部分功能）没有实现。

例如，某商城实现商品优惠活动，如果用户为普通用户，那么商品不打折；如果是商城的会员，那么商品打八五折。代码如下：

```
<?php
     if($user="普通用户"){
      echo $price=485*1;                        //485 是商品价格，1 是指不打折
     }
     if($user="会员"){
      echo $price=485*8.5;                      //485 是商品价格，8.5 是指打八五折
     }
?>
```

上面的代码对于 PHP 编译器来说没有任何问题。运行程序时，程序没有弹出错误信息。但是当用户为商城的会员时，商品价格乘以一个 8.5，这一点就没有符合要求，属于逻辑错误。应该乘以 0.85 才正确。

注意 在实现动态的 Web 编程时，通常情况下，数据表中均是以 8.5 进行存储，这时在程序中就应该再除以 10，这样，就相当于原来的商品价格乘以 0.85。正确的代码为：

```
echo $price=485*8.5/10;                      //485 是商品价格，8.5 是指打八五折
```

对于逻辑错误而言，发现错误是容易的，但要查找出逻辑错误的原因却很困难。因此，在编写程序的过程中，一定要注意使用语句或者函数的书写完整性，否则将导致程序出错。

7.2.4　注释错误

培养编写注释的习惯，对于一个程序员来说是很有帮助的。它可以增强程序的可读性，便于对程序的修改和后期维护。虽然错误的注释并不影响程序的运行，但是会给维护人员在后期的维护上带来一定的难度。例如：

```
<?php
      $backTime=date("Y-m-d",(time()+3600*24*30));     //格式化$backTime 变量为系统当前日期
?>
```

上面获取时间的代码与后面的注释不符，注释应改为“$backTime 为当前日期+30 天期限”。

7.2.5　运行错误

运行错误的原因不容易确定，它可能是由脚本导致的，也可能是在脚本的交互过程中或其他的事件、条件下产生的，这就需要程序员平时多积累遇错处理的方法，以提高解决问题和分析问题的能力。

下面的几种情况是常见的运行错误：

- ☑ 调用不存在的文件。在编写程序时，由于调用文件的名称书写错误，导致调用了一个不存在的文件。
- ☑ 调用不存在的函数。在编写程序时，如果函数名称书写错误，此时就会产生错误。即使函数名写对了，但使用的参数不对，同样也会产生一个错误。
- ☑ 读写文件。访问文件的错误也是经常出现的，如硬盘驱动器出错或写满，人为操作错误导致目录权限改变等。如果没有考虑到文件的权限问题，文件权限设置为只读属性，直接对文件进行操作就会产生错误。由于该文件具有只读的权限，不能进行写入的操作。在执行这项操作时，首先要明确该文件的属性是否为可写。如果要坚持执行操作，那就需要修改文件的权限。
- ☑ 运算的错误。在进行一些算术运算或者逻辑运算的过程中，如果出现不符合运算法则的运算，例如，在做除法运算时，分母为 0，就会产生错误。

7.3　PHP 错误的调试

PHP 最基本的调试策略是使用 PHP 的错误报告机制。之所以说这是最基本的调试策略，是因为对 PHP 文件的设置是在安装时就必须配置好的，否则就无法使用。

7.3.1　PHP 的错误报告

视频讲解：光盘\TM\Video\第 7 章\PHP 的错误报告.exe

通常在一个 PHP 脚本中发生错误时，出错信息会被插入到当前脚本进行输出，如果是致命的错误，脚本就会终止执行。

PHP 将脚本错误分成提示（notice）、警告（warning）和错误（error）3 个等级。

☑ notice 信息可能是脚本运行错误导致的，也可能只是在正常运行过程中出现的。实际上这也许就是代码中的缺陷，因为 PHP 对其解释可能与代码的本义有所不同。

☑ warning 标识着一个非致命性的错误，在代码运行时产生。它们并不是致命的错误，不会停止脚本的执行。

☑ error 消息说明出现了致命的错误，会导致脚本停止运行。PHP 运行的任何阶段都可能产生这种错误，包括初始化、解析和执行代码阶段。

7.3.2　启动错误报告

视频讲解：光盘\TM\Video\第 7 章\启动错误报告.exe

使用 PHP 的错误报告是调试程序的基础。通过对 PHP 配置文件的设置，使程序在执行过程中自动抛出错误的代码行，帮助开发者发现并消除错误。当项目发布后，需要关闭错误报告，用户不能看到错误信息，这样非常不人性化。

启动错误报告，也就是修改 php.ini 文件中的报错设置，具体设置如图 7.5 所示。

display_errors 变量的目的很明显，用于告诉 PHP 是否显示错误。默认值是 Off。但是，要让开发过程更加轻松，需要把这个值设为 On，以便根据错误提示调试程序。

error_reporting 变量的默认值是 E_ALL。这个设置会显示从不良编码实践到无害提示的所有信息。E_ALL 值的设置对于开发过程来讲过于细化。通常只需要设置错误提示和显示不良编码即可，因此，需要把这个值设置为“E_ALL &～E_NOTICE”。

设置完成后，保存 php.ini 文件，然后重新启动 Apache 服务器即可。

通过这些错误设置，开发者在执行程序的过程中，就可以在浏览器中查看到错误信息，以及出错代码的行号，并分析产生错误的原因。

7.3.3　使用 print 语句调试程序

视频讲解：光盘\TM\Video\第 7 章\使用 print 语句调试程序.exe

print 语句用于字符串的输出。基于 print 语句灵活的特点，在程序调试过程中经常被广泛使用。

例 7.5　通过实例介绍如何应用 print 语句对程序进行调试。在本实例中实现一个获取表单提交的数据

功能。**（实例位置：光盘\TM\Instance\07\7.5）**

程序代码如下：

```
<form name="form1" method="post" action="">
  <input name="txt_key" type="text" id="txt_key">
  <input type="submit" name="Submit" value="搜索">
</form>
<?php
if($_POST[Submit]=="搜索"){
     $key=$_POST[txt_key];                          //获取文本框的值
}
?>
```

上面的代码实现了获取文本框的值，由于语句中没有输出，因此当提交表单时也看不到是否获取了信息，这时可以应用 print 语句输出文本框的值，以此来判断是否正确获取文本框的值。在加粗的代码下添加如下语句。

```
print $key;                                         //输出文本框的值
```

print 语句输出了 POST 数组中的数据，即文本框的值，如果数组中的数据未能正确显示，则说明没有正确获取到文本框的值，从而断定代码中有错误，需要查找错误源头，并加以修正。如果 print 语句输出的字符串与输入到文本框中的字符串相符，则说明程序正确。

运行结果如图 7.6 所示。

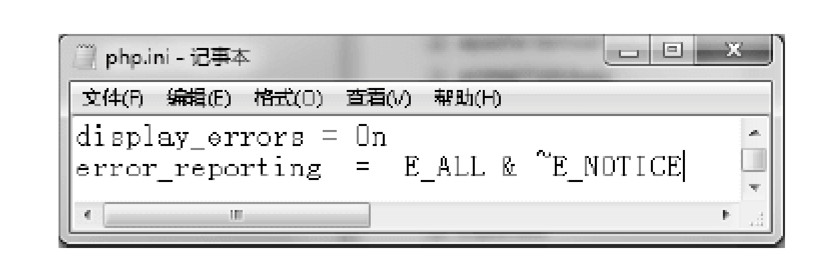


图 7.5　设置 php.ini 文件中的错误处理机制

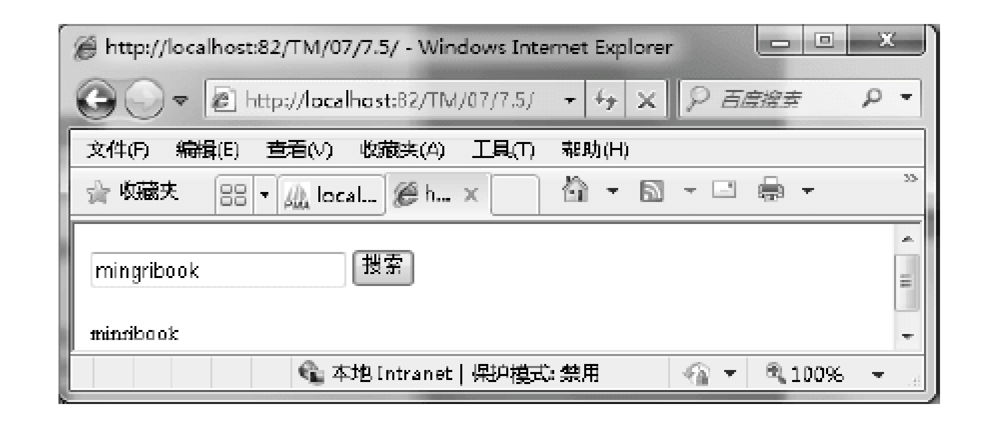


图 7.6　使用 print 语句判断是否正确获取到文本框的值

使用 print 语句可以缩小缺陷可能出现的范围，能够最终确定错误的位置或验证数据信息。在程序中熟练地使用 print 语句，可以迅速找出程序中的错误根源。

7.3.4　应用@前缀字符屏蔽 PHP 脚本错误提示

视频讲解：光盘\TM\Video\第 7 章\应用@前缀字符屏蔽 PHP 脚本错误提示.exe

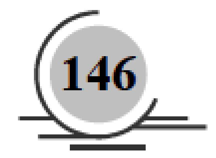
前缀字符@不是一个命令，而是 DOS 批处理的一个特殊标记符，是命令行回显屏蔽符。

前缀字符@表示执行时本行在 cmd 中不显示，可以使用 echo off 命令关闭显示，仅用于屏蔽命令行回显。在默认情况下，当 PHP 代码遇到错误时会显示错误信息，包括当前脚本的绝对路径，这样会造成一些不安全的因素，可以在调用函数时加上前缀字符@，以屏蔽错误信息。

例 7.6　当试图打开一个不存在的文件时，屏蔽错误信息以避免路径泄露。**（实例位置：光盘\TM\Instance\07\7.6）**

代码如下：

```
<?php
$file="book.php";                                   //定义操作文件
@fread($file) or die("文件读取失败！");
fclose($file);
echo "我不能被输出了！程序运行后，前缀字符@后面的命令不被显示！";
?>
```



fread()函数前应用了前缀字符@，说明程序运行时在浏览器上不会显示前缀字符@后面的命令。

结果为：文件读取失败！

7.3.5　使用错误处理器记录日志

PHP 提供了内建的错误处理函数 error_log()，可以将出错信息记录到管理员所指定的路径。

```
error_log(message,type [, destination, [,extra_headers]]);
```

参数 message 指出错信息；参数 type 指定出错信息记录的指定位置。如果用 PHP 的日志记录机制保存出错信息，需要将参数 type 的值设置为 0。如果将错误追加到 destination 文件中，需要将参数 type 的值设置为 3。另外，需要修改 php.ini 中的 error_log 选项，具体设置如图 7.7 所示。



图 7.7　设置 php.ini 文件中的错误处理机制

设置完成后，保存 php.ini 文件，然后重新启动 Apache 服务器即可。

例 7.7　实现一个错误处理的例子，将日志记录到日志文件中，并在文件大于 1024B（1KB）时对日志文件进行重命名。**（实例位置：光盘\TM\Instance\07\7.7）**

代码如下：

```
<?php
function err_log($error,$error_str){                          //自定义一个错误处理函数
	$file="php_error.log";
	if(filesize($file)>1024){                               //如果日志文件大于 1024B
		rename($file,$file.(string)time());             //以时间为依据对日志文件进行重命名
		clearstatcache();                                //清除文件状态缓存
	}
	error_log($error_str,0,$file);                         //将出错信息记录到管理员所指定的路径
}
set_error_handler('err_log');                                  //执行自定义函数 log_roller()
trigger_error(time().":程序报错.\n");                          //发出错误信息
restore_error_handler();                                       //重新编译这个预错处理的函数
?>
```

在上面的代码中，加粗的 PHP 内置错误处理函数 error_log()的 type 参数值设置为 0，将出错信息记录在指定的文件 system logger 中，每运行一次程序或刷新一次，该文件将即时更新，将每次的出错信息加载到该文件中。

技巧　如果将参数 type 的值设置为 3，则将错误追加到 destination 文件中，每运行一次或刷新一次程序，都会生成一个新的错误日志。

通常，内存的地址是以字节（byte）为单位编制的，内存的容量一般以 KB 或 MB 为单位。

一般情况下，在开发一个网站时，为了调试方便，希望错误直接显示在页面上。但当一个网站正式发布到互联网上时，不能把内部的错误信息显示给访问者，最佳的方法是记录到错误日志中。这时，需要在 php.ini 文件中做如下设置。

```
display_errors = Off
log_errors=on
error_log=/tmp/errors.log
```

设置完成后，保存 php.ini 文件，然后重新启动 Apache 服务器即可。

7.4 SQL 错误的调试

SQL 语句错误也是相当常见的，它们是由 MySQL 数据库报告的。SQL 错误主要体现在两个方面：第一是它和 PHP 的连接方面；第二是对其自身的使用上，也就是在执行 MySQL 语句时的错误。

7.4.1 PHP 与 MySQL 连接错误

视频讲解：光盘\TM\Video\第 7 章\PHP 与 MySQL 连接错误.exe

PHP 与 MySQL 数据库的连接是进行程序开发的第一步，如果不能成功地把它们连接起来，工作就无法进行。这就使它们连接中的错误成为首先要解决的问题，也是必须要解决的问题。

在通过 PHP 函数连接 MySQL 数据库时，有可能出现的错误包括：

☑ 使用的 PHP 函数不正确，不过这种情况不是很常见，因为如果是函数书写错误，那么在具备高亮代码显示功能的开发工具中能够直接看出函数书写是否正确。

☑ 使用函数没有问题，但输入的参数不正确。输入了错误的数据库用户名、密码或者指定的数据库不存在，同样会导致数据库连接失败。

例 7.8 通过 PHP 函数连接 db_database07 数据库，数据库用户名为 root，密码为 root。（**实例位置：光盘\TM\Instance\07\7.8**）

代码如下：

```
<?php
  $con=mysql_connect('localhost','root','root') or die("与服务器连接失败!");
  echo "与服务器连接成功!<br>";
  mysql_select_db('db_database07',$con) or die("没有找到数据库!");
  echo "成功!";
?>
```

本实例在编写代码的过程中，使用的密码不是 root，而是 111，所以导致数据库连接失败，其运行结果如图 7.8 所示。

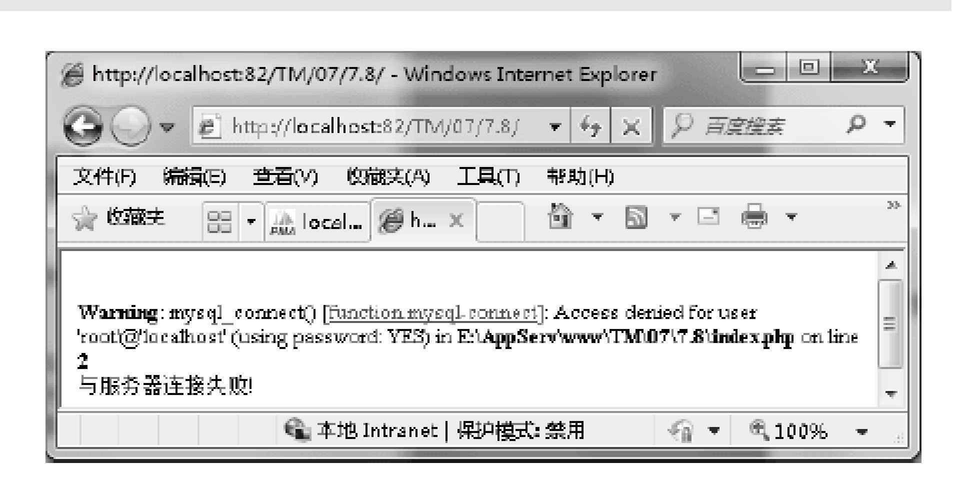


图 7.8 密码不正确导致连接失败

> **注意** 在上述的连接数据库的实例中，应用了一个 die()语句，它是 PHP 的一种错误处理机制。当它左侧的语句发生错误时，die()语句会显示其字符串参数的内容，而且程序会退出。

7.4.2 SQL 语句错误

视频讲解：光盘\TM\Video\第 7 章\SQL 语句错误.exe

编写的 PHP 语句和数据库的连接都没有问题，错误出现在 SQL 语句的执行过程中，出现这种错误的原因有以下 3 个。

☑ 数据提交页面与数据处理页面之间，form 表单中定义的字段的名称与数据处理页中获取字段值所使用的名称不一致，导致获取不到数据，如图 7.9 所示。

☑ 数据处理页中执行的 insert 语句，参数的数量与参数值的数量不匹配，或者参数与参数值的顺序不



匹配，都将导致添加失败，如图 7.10 所示。



图 7.9　数据提交页与数据处理页使用的字段不统一



图 7.10　数据处理页中出现的错误

☑　数据处理页 insert 语句的参数与数据表中字段不匹配，使得字段名称不一致、字段数量不匹配或者数据类型不一致，都将导致添加失败，如图 7.11 所示。

解决 SQL 语句中错误的方法很多，这里介绍 3 种比较常用的方法。

☑　将 SQL 语句复制到对应的数据库中，在数据库图形化管理工具中的 SQL 语言下执行。如果可以执行，那么说明该语句是正确的；如不能执行，则说明是语句本身存在错误，就可以根据数据库图形化管理工具给出的错误信息，分析出错误原因。

☑　使用 mysql_error()语句输出错误信息，通过该语句可以像语法错误那样返回一个错误信息。该语句的使用被放置于 mysql_query()语句的后面。如果将 die()语句与 mysql_error()语句组合应用，当程序结束时就会显示 MySQL 的错误信息。

☑　将 SQL 语句定义成一个单独的变量，应用 echo()语句输出这个变量，分析这个变量输出的值是否正确，如果正确则说明 SQL 语句没有问题。

例 7.9　通过 PHP 函数连接 db_database07 数据库，数据库用户名为 root，密码为 111，向 user 表中添加一条记录，并通过 die()语句与 mysql_error()语句返回错误信息。（**实例位置：光盘\TM\Instance\07\7.9**）

代码如下：

```
<!--处理文件代码-->
<?php
    $conn=mysql_connect('localhost','root','111') or die("与服务器连接失败!");
    mysql_select_db('db_database07',$conn) or die("没有找到数据库!");
    if($Submit=="提交"){
        $admin=$_POST[admin];                                   //获取提交的用户名
        $pass=$_POST[pass];                                     //获取提交的密码
        $dates=date("Y-m-d H:i:s");                             //定义当前时间
        $query="insert into user (admin,password,dates) values('$admin','$pass','$dates')";  //编写添加语句
        $result=mysql_query($query,$conn) or die(mysql_error());       //执行添加语句，并返回错误信息
       if($result){
            echo "<script>alert('添加成功！ '); window.location.href='index.php';</script>";
        }else{
            echo "添加失败!!";
        }
    }
?>
<!--表单提交页，设置的表单元素-->
```

```
<form name="form1" method="post" action="index.php">
        <input name="admin" type="text" id="admin" size="20">
        <input name="pass" type="password" id="pass" size="20">
<input type="submit" name="Submit" value="提交">
</form>
```

本实例中错误的原因是 insert 语句的参数与数据表中的字段名称不匹配，在数据表中没有找到指定的字段，其运行结果如图 7.12 所示。

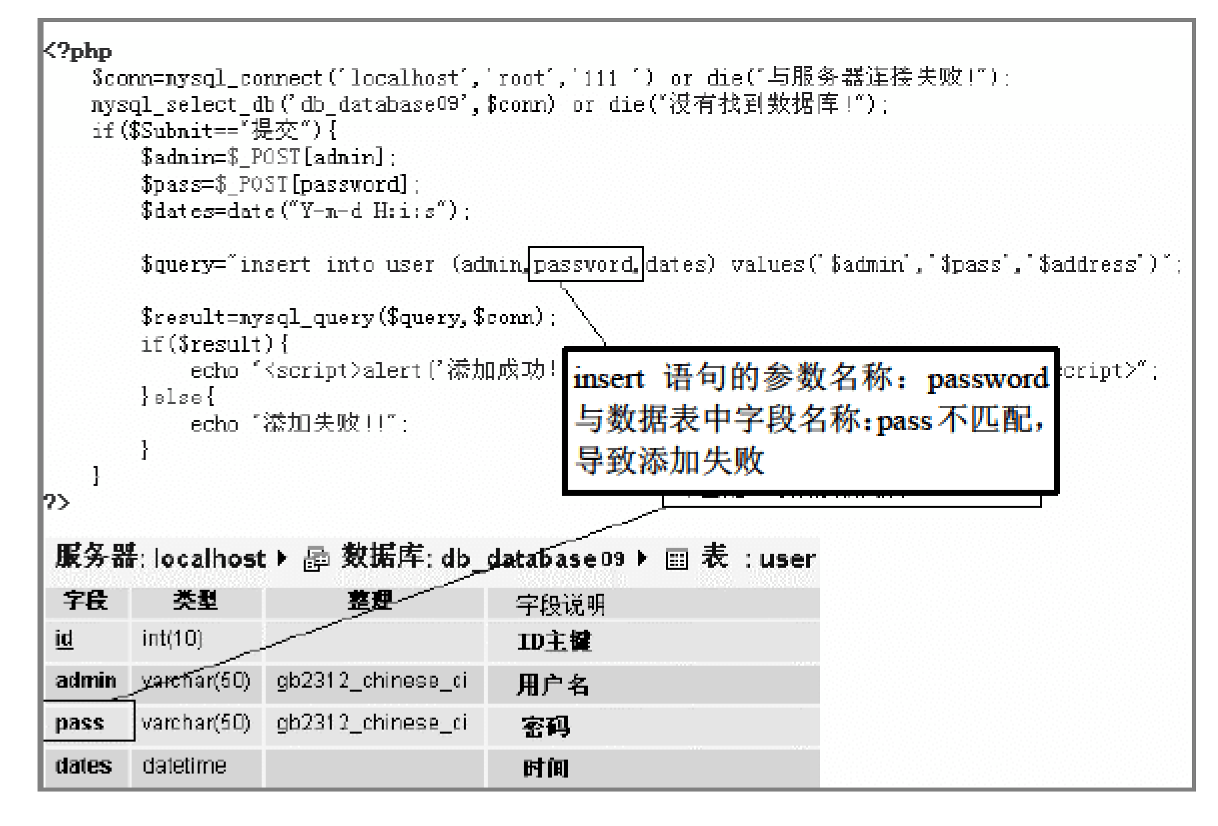


图 7.11　insert 语句的参数与数据表中字段名称不匹配

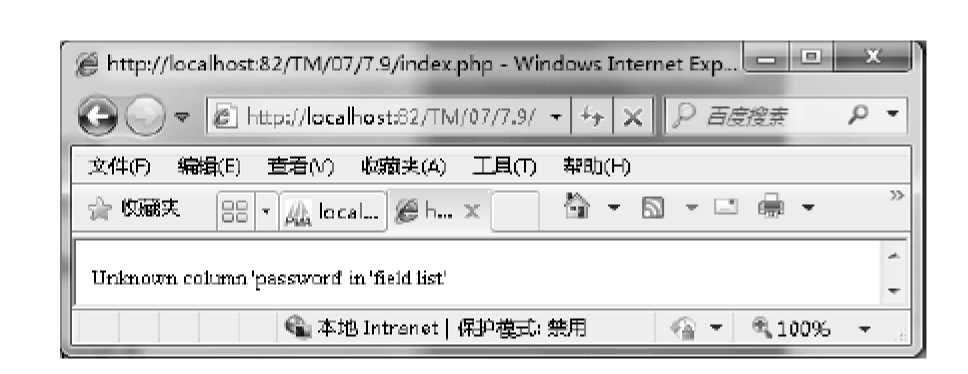


图 7.12　SQL 语句中出现的错误

说明

上面介绍的只是 SQL 语句中的一种错误情况，至于在 select、update 和 delete 等语句的执行过程中的错误需要读者自己去研究与分析。

7.5　实　　战

视频讲解：光盘\TM\Video\第 7 章\实战.exe

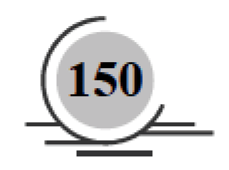
7.5.1　运行缺少第三方组件的程序

在运行光盘中涉及到第三方组件的程序时，由于没有看光盘中提供的使用说明书，而直接运行程序，那么将导致程序运行出错。

例 7.10　在这个实例中，通过 Jpgraph 类库生成折线图，分析 2010 年公司销售额，并且在折线图中通过插入的图像设置数据点标签。但是在直接运行光盘中的程序时，却出现了如图 7.13 所示的错误，其原因就是没有认真的阅读程序使用说明书，因为本程序应用到一个第三方的组件 Jpgraph 类库，需要自行下载，并且将下载的 src 文件夹复制到实例的根目录下，然后才可以运行程序。（**实例位置：光盘\TM\Instance\07\7.10**）

7.5.2　通过 readfile()函数访问远程文件

例 7.11　在本实例中通过文件系统函数库中的 readfile()函数访问一个远程文件，但是由于本机 php.ini



文件的配置，不允许访问远程文件，结果导致程序运行失败。**（实例位置：光盘\TM\Instance\07\7.11）**

代码如下：

```
<?php
	if(@readfile('http://192.168.1.235/','r')){		//判断文件打开的操作是否执行成功
	}else{
	echo "<script>alert('您没有访问远程文件的权限！');window.location.href='error.html'</script>";
								//重定向页面
}
?>
```

如果想让服务器支持远程文件的访问，必须修改 php.ini 文件，定位到 allow_url_fopen = OFF 选项，将 allow_url_fopen 的值修改为 ON，保存后重新启动 Apache 服务器即可。

运行结果如图 7.14 所示。

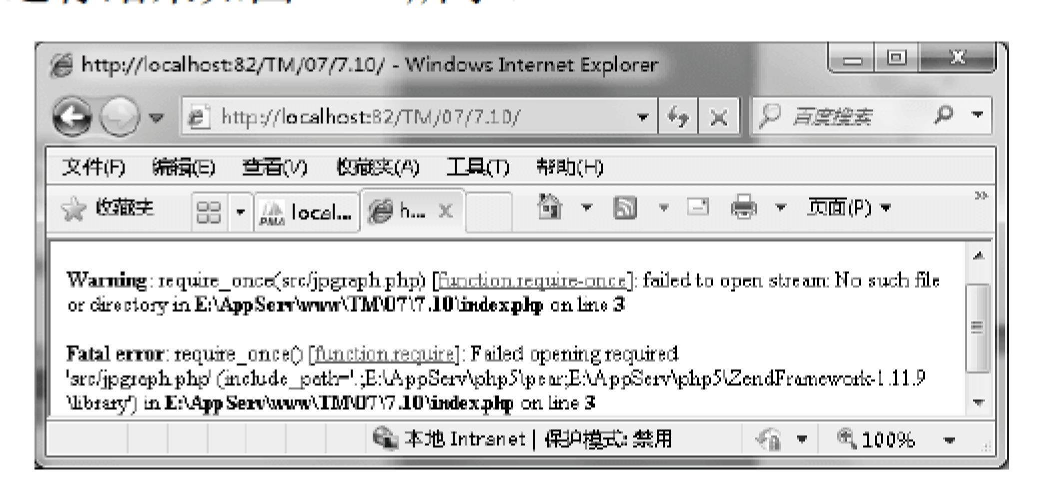


图 7.13　没有找到指定的包含文件

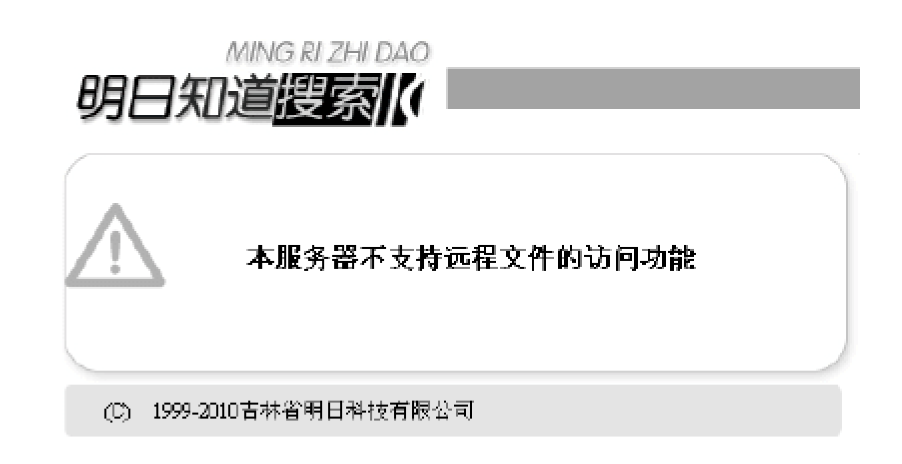


图 7.14　服务器不支持远程文件的访问

7.5.3　解决数据库乱码问题

在通过 PHP 对 MySQL 数据库中的数据进行操作的过程中，有时输出 MySQL 数据库中的数据会出现乱码。这是一个让人十分头疼的问题，纠其根源该问题出现的原因是数据库中数据对字符编码格式的限制。由于在创建数据库时使用的编码格式是“utf8”，所以在对数据库中的数据进行输出或者添加过程中必须使用相同的编码格式才行，否则就出现乱码。

例 7.12　在本实例中，通过 while 语句循环输出数据库中数据，比较在设置数据库编码格式与不设置数据库编码格式时输出数据的不同之处。**（实例位置：光盘\TM\Instance\07\7.12）**

（1）创建数据库连接文件 conn.php，完成与 MySQL 服务器的连接，用户名是 root，密码为 111。然后，连接 db_database07 数据库。最后设置数据库编码格式为 utf8。其关键代码如下：

```
<?php
$conn=mysql_connect("localhost","root","111") or die("服务器连接失败：".mysql_error());	//连接服务器
mysql_select_db("db_database07",$conn) or die ("数据库连接失败：".mysql_error());	//连接数据库
mysql_query("set names utf8");						//设置数据库编码格式
?>
```

（2）创建 index.php 文件。首先通过 include_once()语句包含数据库连接文件，然后定义 SQL 查询语句，最后执行 SQL 语句，并且通过 while 语句完成数据库中数据的循环输出。其关键代码如下：

```
<?php
		include_once("conn/conn.php");				//包含数据库连接文件
		$sql="select * from tb_log ";				//定义 SQL 语句
		$query=mysql_query($sql,$conn);				//执行查询操作
		echo mysql_error();					//返回错误信息，如果存在
		while($myrow=mysql_fetch_array($query)){		//循环输出查询结果
		?>
	<tr>
	  <td><?php echo $myrow['id']?></td>
```

```
                    <td><?php echo $myrow['log_name']?></td>
                    <td><?php echo $myrow['log_date'];?></td>
                </tr>
        <?php
                }
?>
```

运行效果如图 7.15 和图 7.16 所示。

ID	标题	时间
1	数据库服务器问题	2012-11-12 08:31:49
2	数据表问题	2012-11-12 08:32:23
5	复方法	2012-11-14 07:52:57

图 7.15　正常的数据输出结果

ID	标题	时间
1	????????	2012-11-12 08:31:49
2	?????	2012-11-12 08:32:23
5	???	2012-11-14 07:52:57

图 7.16　数据输出乱码

7.5.4　封装异常处理类

例 7.13　在程序调试中，可以应用 try…catch{}语句捕获错误信息，并且通过 Exception 内置类来返回相应的错误信息。在此基础上还可以使用继承对 Exception 类进行扩展，编写属于自己的异常处理类。其方法就是，编写一个子类来继承 Exception 类，这样在子类中就继承了父类的所有属性和方法，并且还可以添加子类所特有的属性和方法。在本项目中编写一个可以判断电话号码格式是否正确的类，当定义的电话号码格式不正确时跳转到自定义的错误页面，并且输出错误提示信息。**（实例位置：光盘\TM\Instance\07\7.13）**

（1）封装电话号码格式判断的异常处理类 TelException，继承 Exception 类。其关键代码如下：

```
class TelException extends Exception{                    //定义 TelException 类，继承 Exception 类
        public function errorTel(){                      //定义方法返回错误信息
            $errorMsg = "出错原因：".$this->getMessage()."不是一个合法的电话号码";
            $errorMsg .="<br>";
            $errorMsg .="错误文件路径：".$this->getFile();
            $errorMsg .="<br>";
            $errorMsg .="错误代码行号：".$this-> getLine();
            return $errorMsg;
        }
}
```

（2）创建自定义函数 check_tel()，通过正则表达式和 preg_match()函数验证电话号码的格式是否正确。自定义函数的语法如下：

```
function check_tel($tel){                                //自定义函数验证电话号码格式是否正确
    $checkphone="/^13(\\d{9})$/";                        //定义验证手机号码的正则表达式
    $counts=preg_match($checkphone,$tel);                //执行验证操作
    return $counts;                                      //返回验证结果
}
```

（3）定义被验证的电话号码，应用自定义异常处理类对电话号码格式进行验证。其代码如下：

```
$tel = "1371234****";                                    //定义被验证的电话号码
/*
    通过自定义异常处理类返回错误提示
*/
try {
    if(check_tel($tel) !=1){
        throw new TelException($tel);
    }
}catch (TelException $e){
    include_once("error.php");
}
```

其运行结果如图 7.17 所示。



图 7.17　判断电话号码的格式是否正确

7.5.5　解决程序的语法错误

在应用 if、for、while 和 switch 等流程控制语句时，必须注意其中大括号的书写，它们必须是成对出现的，否则将导致程序运行出错。**（实例位置：光盘\TM\Instance\07\7.14）**

例 7.14　在本实例中，实现一个简单的网页计数器的功能，并且将数据存储到指定的文本文件中。其关键代码如下：

```
<?php
if(($fp=fopen("counter.txt","r"))==false){
	echo "打开文件失败!";
}else{
	$counter=fgets($fp,1024);                    //读取文件中数据
	fclose($fp);                                 //关闭文本文件
	$counter++;                                  //计数器增加 1
	$fp=fopen("counter.txt","w");                //以写的方式打开文本文件
	fputs($fp,$counter);                         //将新的统计数据增加 1
	fclose($fp);
if(($fp=fopen("counter.txt","r"))==false){       //打开文件
	echo "打开文件失败!";
}else{
	$counter=fgets($fp,1024);                    //读取文件中数据
	fclose($fp);                                 //关闭文件
echo $counter;
}
?>
```

运行效果如图 7.18 所示。

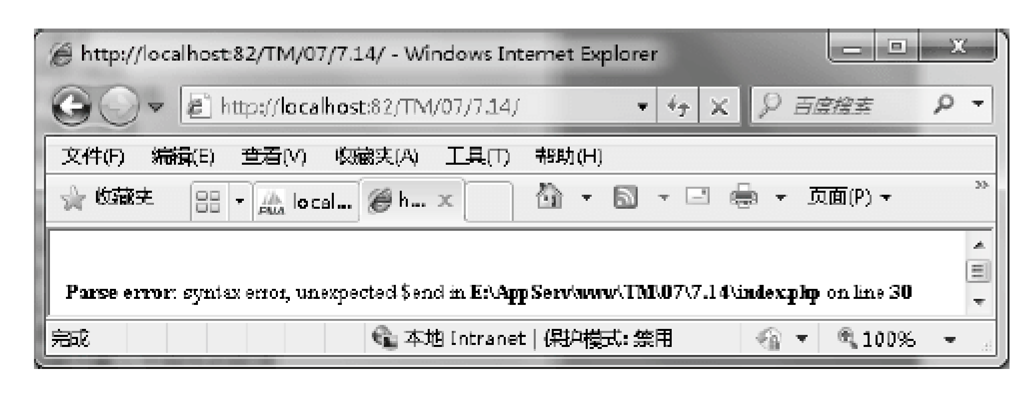


图 7.18　由于大括号不完整导致的错误

7.6　小　　结

本章主要介绍了程序调试及异常处理的一般步骤，并着重讲解了 PHP 中常见的错误及其原因。通过本

章的学习，初学者可以掌握 PHP 中常见错误的处理方式，并能够积累基本的程序调试经验与技巧。希望读者朋友能够以本章介绍的内容为基石，进一步拓展与积累，最终登上编程工程师的舞台。

7.7 学习成果检验

1．连接不存在的数据库，在运行光盘中涉及到数据库的程序时，经常会犯这样的一个错误，就是没有进行数据库的附加，而直接运行程序，则导致输出错误信息。**（实例位置：光盘\TM\Instance\07 \7.15）**

2．在开发网站时，为了调试方便，将错误直接显示在页面上。但当一个网站正式发布到互联网上时，就不能把内部的错误信息显示给访问者，其最佳的方法应该是记录到错误日志中。实现一个错误处理的例子，将日志记录到日志文件中，并在文件大于 1024B（1KB）时对日志文件进行重命名。**（实例位置：光盘\TM\Instance\07\7.16）**

注意 PHP 提供内建的错误处理函数 error_log()，可以将出错信息记录到管理员所指定的路径。error_log()函数的语法如下：

```
error_log(message,type [, destination, [,extra_headers]]);
```

参数 message 指出错信息；参数 type 指定出错信息记录的指定位置。如果用 PHP 的日志记录机制保存出错信息，需要将参数 type 的值设置为 0；如果将错误追加到指定的文件中，需要将参数 type 的值设置为 3。

将错误日志存储到指定文件中，不在页面中显示，还需要对 php.ini 文件做如下设置：

```
display_errors = Off
```

设置完成后，保存 php.ini 文件，然后重新启动 Apache 服务器即可。

第 8 章

综合实例（一）——在线论坛

（视频讲解：25分钟）

论坛系统已经成为时下比较普遍的一种网上交流手段，现在很多网站都拥有自己的论坛，特别是对于大型的网站，论坛系统已经成为一个不可缺少的组成部分。本章将详细介绍论坛系统的开发、基本操作流程和创作思路，为读者提供一个整体的开发思路。

通过阅读本章内容，你可以：

- 熟悉用户注册模块的设计
- 熟悉用户登录模块的设计
- 熟悉帖子分类管理模块的设计
- 了解发帖模块设计
- 了解回帖模块设计
- 熟悉后台管理模块设计

8.1　在线论坛概述

视频讲解：光盘\TM\Video\第 8 章\在线论坛.exe

本章开发的是一个最基本、最简单的论坛系统，其具备了论坛系统的基本功能，没有附加任何复杂的功能，完全适合初学者学习和研究。论坛系统的具体功能如下：

☑ MySQL 数据库的创建。
☑ 用户注册。
☑ 用户登录。
☑ 帖子的分类管理。
☑ 发布帖子。
☑ 回复帖子。
☑ 后台管理。

8.1.1　程序业务流程

在线论坛系统的操作流程非常清晰，总体上由两大模块组成：前台展示区和后台管理，其中前台展示区的主要功能包括用户注册、用户登录、发布帖子、回复帖子等；后台管理模块的主要功能包括用户管理、栏目管理、主题管理、回复内容管理和非法信息管理。程序流程如图 8.1 所示。

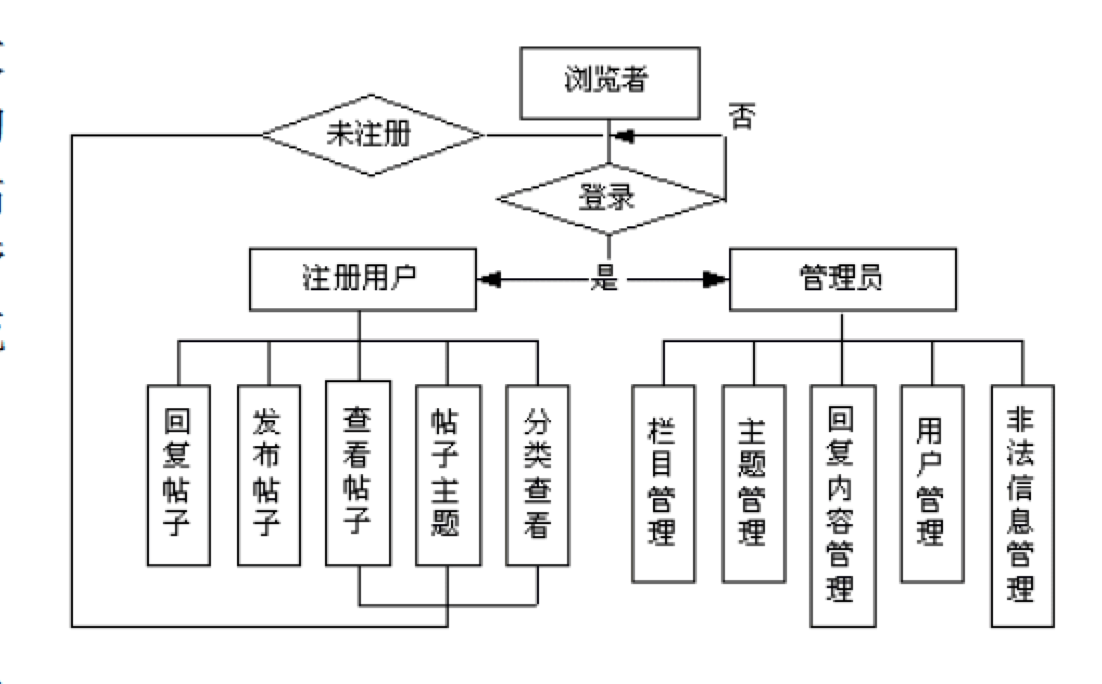


图 8.1　在线论坛系统操作流程图

8.1.2　系统预览

在线论坛由多个页面组成，下面仅列出几个典型页面，其他页面参见光盘中的源程序。

论坛首页面的运行效果如图 8.2 所示。

用户注册页面的运行效果如图 8.3 所示，主要功能是实现用户注册。



图 8.2　论坛首页运行效果

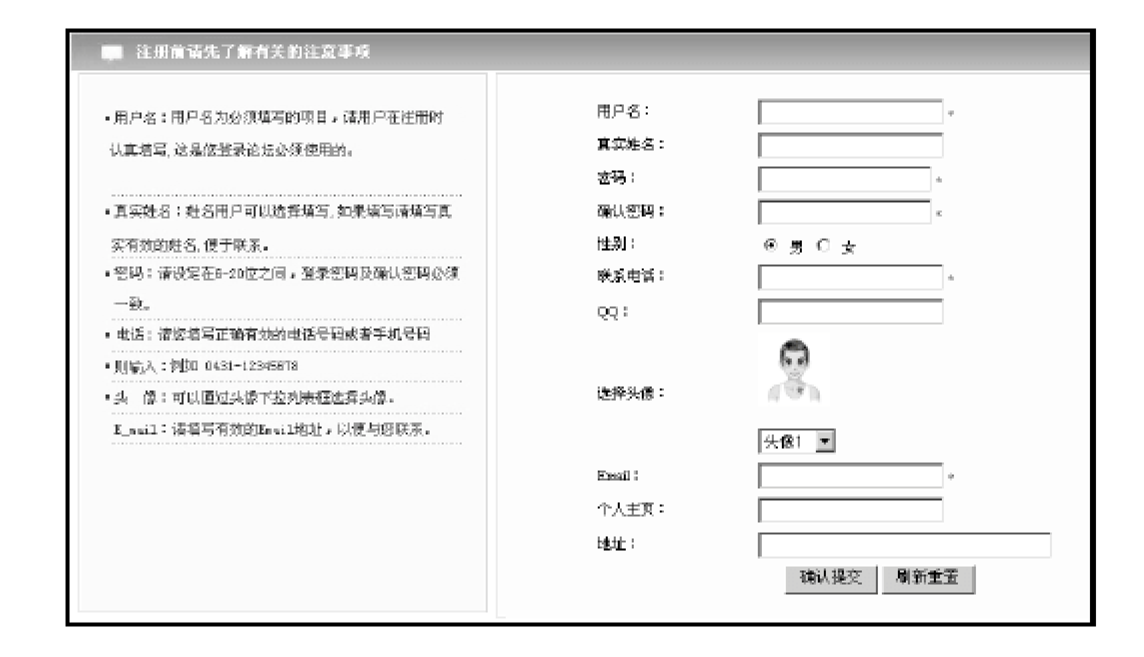


图 8.3　用户注册页面运行效果

发布主题页面的运行效果如图 8.4 所示，主要功能是实现发布帖子。

后台首页的运行效果如图 8.5 所示。

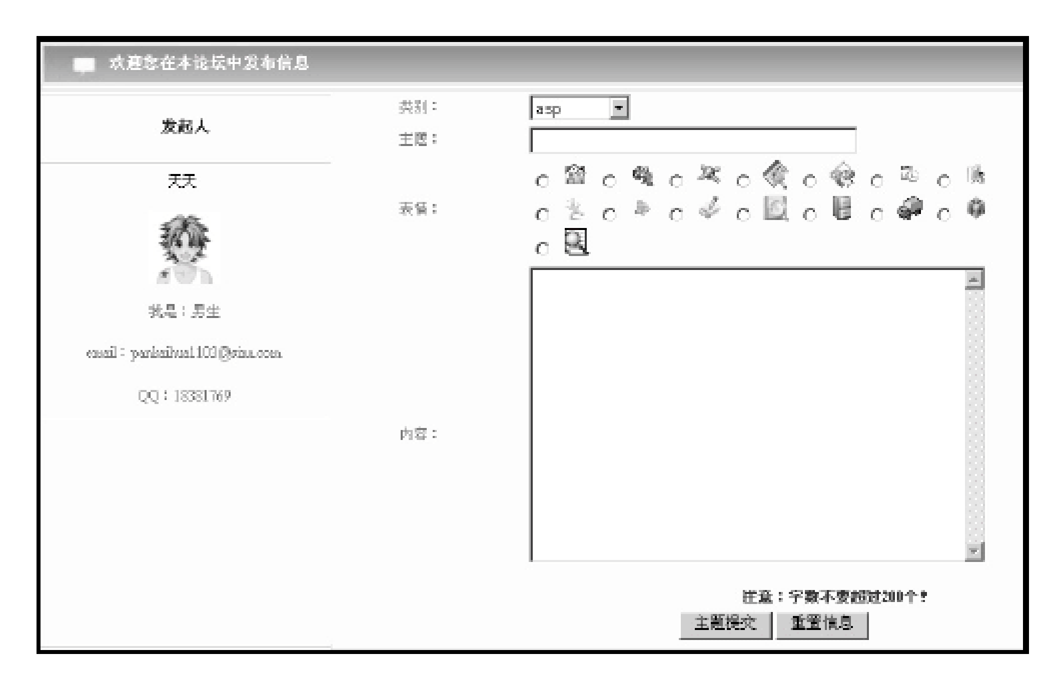


图 8.4　发布主题页面运行效果



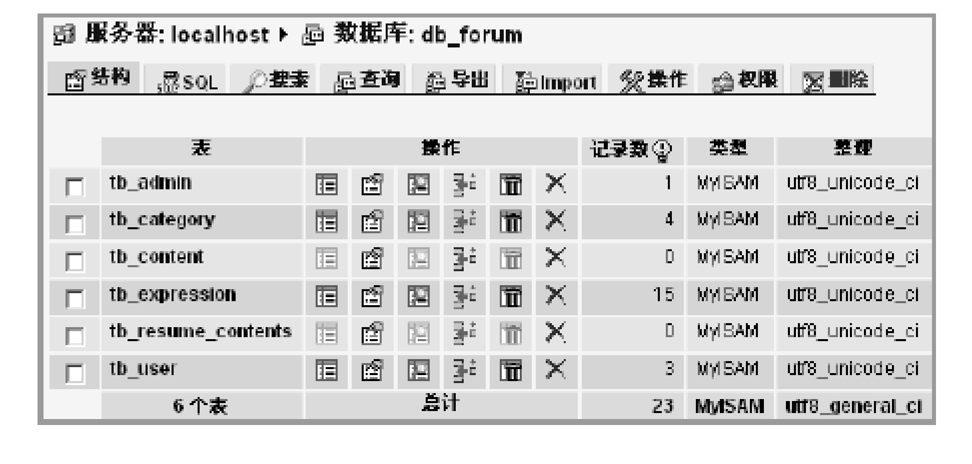
图 8.5　后台首页运行效果

8.2　数据库设计

视频讲解：光盘\TM\Video\第 8 章\数据库设计.exe

8.2.1　数据库概要说明

在线论坛系统中，采用的是 MySQL 数据库，用来存储用户信息、发帖信息、回帖信息等。这里将数据库命名为 db_forum，其中包含的数据表如图 8.6 所示。



服务器: localhost ▸ 数据库: db_forum

结构　SQL　搜索　查询　导出　Import　操作　权限　删除

表	操作	记录数	类型	整理
tb_admin		1	MyISAM	utf8_unicode_ci
tb_category		4	MyISAM	utf8_unicode_ci
tb_content		0	MyISAM	utf8_unicode_ci
tb_expression		15	MyISAM	utf8_unicode_ci
tb_resume_contents		0	MyISAM	utf8_unicode_ci
tb_user		3	MyISAM	utf8_unicode_ci
6 个表	总计	23	MyISAM	utf8_general_ci

图 8.6　数据库结构

8.2.2　数据库概念设计

根据业务流程和系统功能结构，规划出系统中使用的数据库实体对象及实体 E-R 图。在创建数据表前，首先需要创建基本信息的数据表，如图像信息表、版主信息表、管理员信息表等。

图像信息表实体 E-R 图如图 8.7 所示。

版主信息表实体 E-R 图如图 8.8 所示。

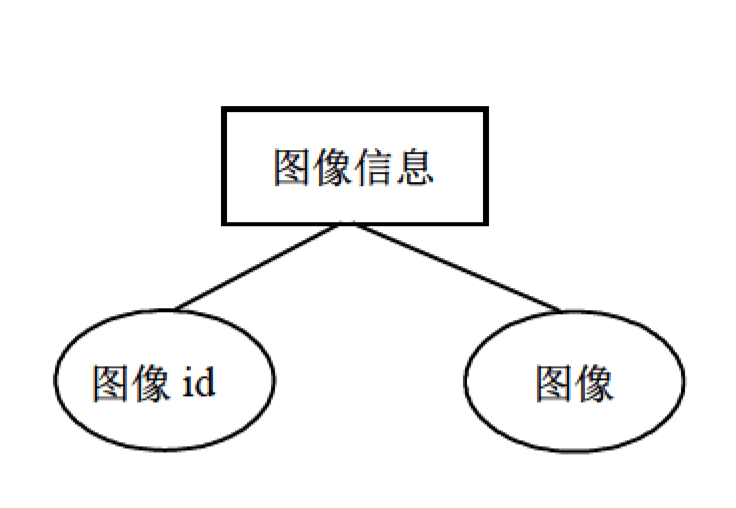


图 8.7　图像信息表实体 E-R 图

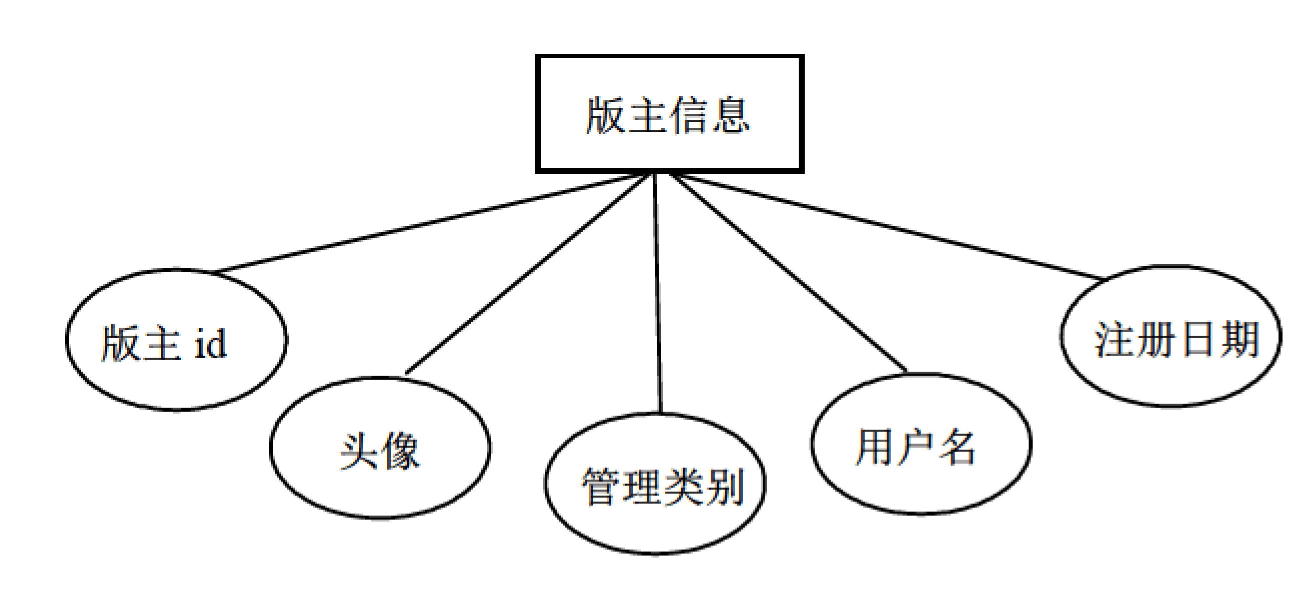
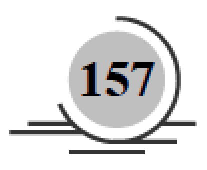


图 8.8　版主信息表实体 E-R 图

管理员信息表实体 E-R 图如图 8.9 所示。

当用户注册后，需要将用户信息存储到数据库中，包括用户的用户名、真实姓名、密码、住址等属性，其实体 E-R 图如图 8.10 所示。



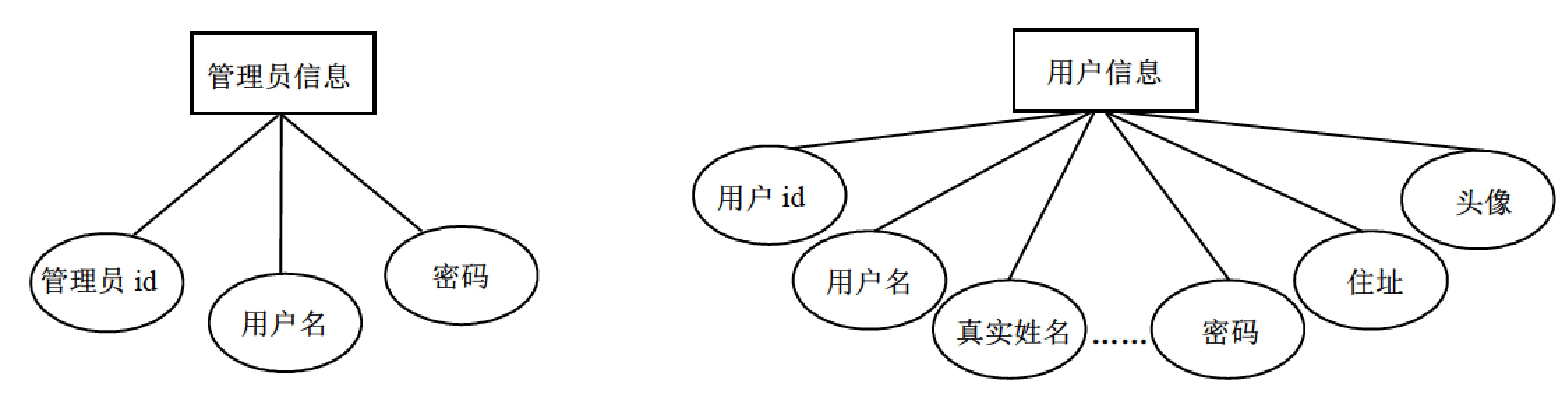


图 8.9　管理员信息表实体 E-R 图　　　　图 8.10　用户信息表实体 E-R 图

当用户发布帖子信息时，需要将帖子信息存储到数据库中，包括帖子类别、帖子主题、帖子内容等，其实体 E-R 图如图 8.11 所示。

当用户回帖时，需要将回帖信息存储到回帖表中，包括回帖主题、回帖内容、回帖人、原帖主题等，其实体 E-R 图如图 8.12 所示。

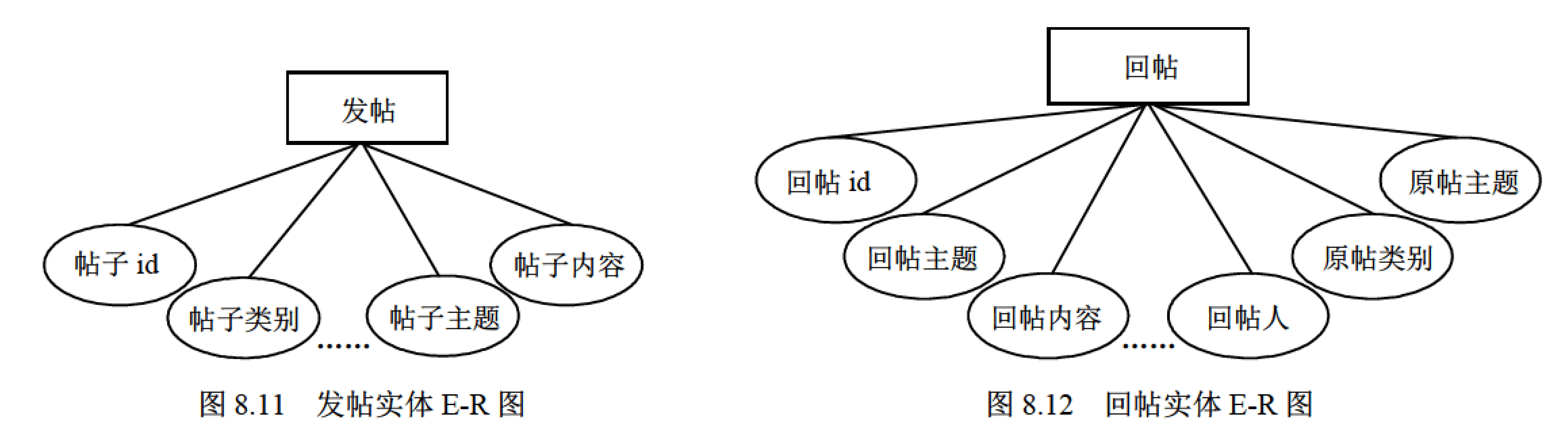


图 8.11　发帖实体 E-R 图　　　　图 8.12　回帖实体 E-R 图

8.2.3　数据库逻辑设计

在线论坛系统是典型的数据库开发应用程序，论坛的数据库设计是一个非常关键的环节，下面将对本论坛系统中使用的数据库进行介绍。

论坛系统中创建的数据库名称是 db_forum，MySQL 数据库服务器的用户名是 root，密码是 81。在创建的数据库中包括 6 个数据表，其中各数据表实现的功能如表 8.1 所示。

表 8.1　db_forum 数据库中数据表功能说明

数据表名称	功 能 说 明
tb_admin	管理员信息表，存储管理员的个人信息
tb_category	论坛栏目信息表，存储论坛中创建的栏目信息
tb_content	发布帖子信息表，存储用户在论坛中发布的帖子信息
tb_expression	表情图存储表，存储在论坛中使用的表情图
tb_resume_contents	回复帖子信息表，存储对论坛中帖子的回复内容
tb_user	注册用户的个人信息表，存储注册用户的个人信息

注意 在创建数据库的过程中一定要注意字符集的使用，要选择使用 utf-8 类型，如果使用其他字符集，有可能会导致数据库中的数据出现乱码。

这里使用了 6 个数据表，其中各个表的结构不再一一介绍。下面以用户信息表为例，来了解一下数据表的创建过程，以及其中需要注意的问题。

tb_user 用户信息表的结构如图 8.13 所示。

服务器: localhost ▸ 数据库: db_forum ▸ 表 : tb_user

浏览 | 结构 | SQL | 搜索 | 插入 | 导出 | Import | 操作 | 清空 | 删除

字段	类型	整理	属性	Null	默认	额外
id	int(10)			否		auto_increment
username	varchar(50)	utf8_unicode_ci		否		
true_name	varchar(50)	utf8_unicode_ci		是	NULL	
password	varchar(50)	utf8_unicode_ci		否		
sex	varchar(20)	utf8_unicode_ci		是	NULL	
tel	varchar(20)	utf8_unicode_ci		否		
email	varchar(50)	utf8_unicode_ci		否		
qq	varchar(20)	utf8_unicode_ci		是	NULL	
indexs	varchar(50)	utf8_unicode_ci		是	NULL	
address	mediumtext	utf8_unicode_ci		是	NULL	
tx	varchar(50)	utf8_unicode_ci		是	NULL	

图 8.13　tb_user 用户信息表的结构

在使用 MySQL 数据库创建数据表时，首先要指定一个字段为数据表主键，其类型为 int（如用户表中的 id），然后在创建其他字段时，要根据字段表述的内容为字段定义类型，例如，表示时间的字段可以使用 date 或者 datetime 等时间类型，而表述大量的文本字段时，应该使用 text 类型，如果存储的是二进制数据，那就要定义 blob 或者 longblob 类型，具体字段使用什么样的类型来定义，要根据具体问题具体分析。

注意 在创建数据表时，一定要指定数据表的类型为 MyISAM，如果使用其他类型将影响数据库的备份，例如，使用 InnoDB 类型保存数据，如果将表中的数据复制到其他机器的数据库中，该数据表将不可用。

8.3　用户注册模块设计

视频讲解：光盘\TM\Video\第 8 章\用户注册模块设计.exe

8.3.1　用户注册模块概述

当用户第一次登录本论坛时，必须先进行注册，然后登录，才可以发表帖子。单击“注册”按钮，进入到注册页面，按照要求输入注册信息，单击“确认提交”按钮，即可成功注册账号。用户注册模块运行效果如图 8.14 所示。



图 8.14　用户注册模块的运行结果

8.3.2 JavaScript 脚本和 include()包含语句

1. JavaScript 脚本的应用

在论坛的用户注册模块中，主要表述的是用户注册信息，所以在进行页面设计时，重点把握如何设计表单元素的输出方式，以及如何对表单中提交的数据进行处理。在对表单提交的数据进行处理的过程中，使用 JavaScript 脚本和正则表达式对表单中提交的数据进行判断，以及对用户头像进行选择。主要代码如下：

```
<script language="javascript">//通过下拉列表选择头像时应用该函数
    function showlogo(){
            document.images.img.src="images/tx/"+
     document.form1.tx.options[document.form1.tx.selectedIndex].value;
     }
</script>
```

2. include()包含语句的应用

为了便于对论坛中头尾文件进行修改，这里充分发挥了 include()包含语句的作用，头尾文件都是通过包含语句直接调用的，无须在本页编辑。

（1）应用 include 语句包含数据库服务器连接文件 conn.php、论坛的头文件 index_01.php 和论坛的用户登录文件 index_02.php。代码如下：

```
<?php
include("conn/conn.php");
include("index_01.php");
include("index_02.php");
?>
```

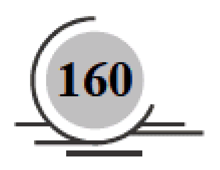
（2）将美工设计的网页效果图嵌入用户注册信息页面，应用表格技术合理地划分和设计表单的布局，在对应的表格中嵌入表单元素，并且设置表单元素的名称。

注意 由于用户注册页面的设计中涉及的表单元素较多，所以一定要注意表单元素名称的使用，其中设计的名称一定要与表单处理页中使用的表单元素的变量名称相吻合，否则将导致上传数据失败。

用户注册模块中涉及到的 HTML 表单元素如表 8.2 所示。

表 8.2 用户注册页面涉及到的 HTML 表单元素

名　称	类　型	含　义	重 要 属 性
form 1	form	表单	method="post" action="login_ok.php" onSubmit="javascript: return checkit();"
username	text	注册用户名	<input name="username" type="text" id="username" />
true_name	text	真实姓名	<input name="true_name" type="text" id="true_name" />
zc_password	password	注册密码	<input name="password" type="password" id="password" /> *
password2	password	确认密码	<input name="password2" type="password" id="password2" /> *
sex	radio	性别	<input name="sex" type="radio" value="男" checked="checked" /> 男 <input type="radio" name="sex" value="女" /> 女
tel	hidden	电话	<input name="tel" type="text" id="tel" />
qq	submit	QQ 号码	



续表

名　　称	类　　型	含　　义	重 要 属 性
tx	select	头像	<select size"1" id="tx" name="tx" onChange="showlogo()">
email	Text	邮箱地址	
indexs	Text	个人主页	
address	Text	联系地址	
Submit	Submit	提交表单	
Submit2	reset	重置表单	

8.3.3　用户注册模块的实现过程

1. 页面设计

本实例中设计的用户注册模块包括 3 部分内容：论坛的头文件、论坛中表单元素的详细设置和论坛中尾文件的设置。用户注册模块的设计效果如图 8.15 所示。



图 8.15　用户注册模块的设计效果

2. 代码设计

在完成用户注册模块的整体布局设计后，接着实现用户注册模块的功能。首先通过 JavaScript 脚本对表单中提交的数据进行判断。关键代码如下：

（代码位置：光盘\TM\Instance\08\login.php）

```
<script language="javascript">
function checkit(){
            if(form1.username.value==""){
                alert("请输入用户名!");
                 form1.username.select();
                  return(false);
            }
            if(form1.password.value==""){
                alert("请输入用户密码!");
                  form1.password.select();
                  return(false);
             }
    ……//省略部分代码
          if(!checkemail(form1.email.value)){
                  alert("邮箱地址格式不正确!");
                  form1.email.select();
                  return(false);
```

```
            }
                return(true);
}
function checkemail(email){                              //验证邮箱地址格式是否正确
        var strs=email;
        var Expression=/\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*/;
        var objExp=new RegExp(Expression);
        if(objExp.test(strs)==true){
                return true;
        }else{
                return false;
        }
}
function checkphone(tel){                                //验证电话号码格式是否正确
        var str=tel;
        var Expression=/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/;
        var objExp=new RegExp(Expression);
        if(objExp.test(str)==true){
                return true;
        }else{
                return false;
        }
}
</script>
```

当用户单击头像下拉列表选择头像时，显示头像图片。程序代码如下：

（代码位置：光盘\TM\Instance\08\login.php）

```
<script language="javascript">                           //通过下拉列表选择头像时应用该函数
       function showlogo(){
                document.images.img.src="images/tx/"+
        document.form1.tx.options[document.form1.tx.selectedIndex].value;
        }
</script>
              <tr>
                <td>选择头像：</td>
                <td><p><img src="images/tx/1.gif" id="img" name="img" width="60" height="60" /></p>
                    <select size"1" id="tx" name="tx" onChange="showlogo()">
                      <option value="1.gif">头像 1</option>
                        <option value="2.gif">头像 2</option>
                        <option value="3.gif">头像 3</option>
                        <option value="4.gif">头像 4</option>
                        <option value="11.gif">头像 11</option>
                        <option value="6.gif">头像 6</option>
                        <option value="7.gif">头像 7</option>
                        <option value="8.gif">头像 8</option>
                        <option value="9.gif">头像 9</option>
                        <option value="10.gif">头像 10</option>
                    </select>
```

当用户单击“确认提交”按钮后，将表单中的数据提交到用户注册信息处理页 login_ok.php 中，再将数据添加到数据库中进行存储。程序代码如下：

（代码位置：光盘\TM\Instance\08\login_ok.php）

```
<?php
include("conn/conn.php");
if(isset($_POST['Submit']) and $_POST['Submit']=="确认提交"){
        $username=$_POST['username'];
        $true_name=$_POST['true_name'];
        $password=$_POST['password'];
        $sex=$_POST['sex'];
```

```
        $tel=$_POST['tel'];
        $email=$_POST['email'];
        $qq=$_POST['QQ'];
        $indexs=$_POST['indexs'];
        $address=$_POST['address'];
        $tx="images/tx/".$_POST['tx'];
         if($_POST['password']==$_POST['password2']){
                $insert=mysql_query("insert into tb_user(username,true_name,password,sex,tel,email,qq,indexs,address,tx)
values('$username','$true_name','$password','$sex','$tel','$email','$qq','$indexs','$address','$tx')",$conn);
                if($insert){
                        echo "<script>alert('注册成功！');window.location.href='index.php';</script>;";
                }else{
                        echo "<script>alert('注册失败！');window.location.href='index.php';</script>;";
                }
        }else{
                echo "<script>alert('两次输入的密码不一致！');window.location.href='login.php';</script>;";
}
}
?>
```

8.4　用户登录模块设计

视频讲解：光盘\TM\Video\第 8 章\用户登录模块设计.exe

8.4.1　用户登录模块概述

用户登录模块是用户进入到程序系统的门户，通过登录模块，可以对登录用户进行身份验证，只有系统的合法用户才可以进入系统的主界面。整个登录模块的实现过程非常简单，相信读者会很快掌握。登录模块的运行效果如图 8.16 所示。

8.4.2　通过 JavaScript 脚本判断用户名和密码是否为空

在用户登录模块中，当用户单击“登录”按钮时，使用 JavaScript 脚本技术判断用户名和密码是否为空，如果为空则给出提示。具体代码如下：

```
<script language="javascript">
function check(){                                    //自定义函数
        if(form3.user.value==""){                    //判断用户名是否为空
                alert("请输入用户名!");
                 form3.user.focus();
                  return false;
            }
          if(form3.pwd.value==""){                   //判断密码是否为空
                  alert("请输入密码!");
                  form3.pwd.focus();
                  return false ;
              }
                  return true;
}
</script>
```

8.4.3 用户登录模块的实现过程

1. 页面设计

用户登录模块的设计非常简单，只包括两个表单元素和一个提交按钮，虽然内容少，但对其与其他内容进行合理的搭配也是非常重要的，在本实例中设计的用户登录模块的效果如图 8.17 所示。

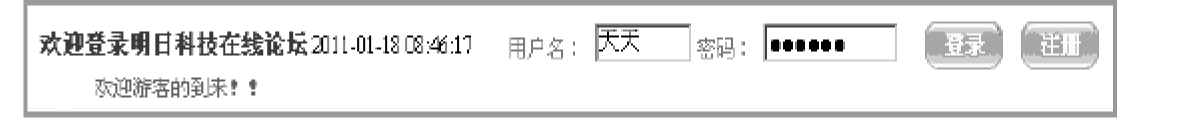


图 8.16 用户登录模块的运行效果

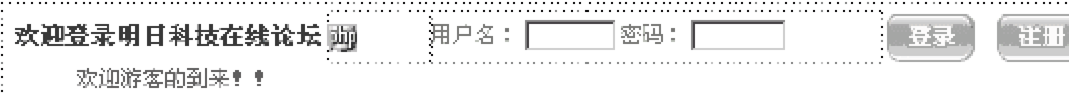


图 8.17 用户登录模块的效果

用户登录模块的页面设计流程如下。

（1）输出系统的当前时间，代码如下：

```
<?php echo date("Y-m-d H:i:s");?>
```

（2）设计用户登录添加的表单元素和按钮。

用户登录模块中涉及到的 HTML 表单元素如表 8.3 所示。

表 8.3 用户登录页面涉及到的 HTML 表单的重要元素

名 称	类 型	含 义	重 要 属 性
form 3	form	表单	method="post" action="user.php"
user	text	登录用户名	
pwd	password	密码	
imageField	Submit	登录按钮	<input type="image" name="imageField2" src="images/02_011.gif" onclick= "return check();" />

2. 代码设计

用户登录模块的功能主要通过两个步骤来完成。首先在用户登录页面中输入用户名和密码，然后单击“登录”按钮，将数据提交到用户登录数据处理页中，对提交的数据进行验证，如果正确则提示用户登录成功，否则返回到用户登录页面。其中用户登录数据处理页中的代码如下：

（代码位置：光盘\TM\Instance\08\user.php）

```
<?php
session_start();                                //初始化 session 变量
?>
<?php
include("conn/conn.php");                       //包含数据库连接文件
if(isset($_POST['user']) and isset($_POST['pwd'])){
$select=mysql_query("select * from tb_user where username='".$_POST['user']."' and password='".$_POST['pwd']."'",
$conn);
if(mysql_num_rows($select)==1){
        $array=mysql_fetch_array($select);
        $_SESSION['user']=$_POST['user'];
        $_SESSION['sex']=$array['sex'];
        $_SESSION['email']=$array['email'];
        $_SESSION['qq']=$array['qq'];
        $_SESSION['tx']=$array['tx'];
        echo "<script>alert('登录成功！');window.location.href='index.php'</script>;";
}else{
        echo "<script>alert('登录失败！');window.location.href='index.php'</script>;";
}
}
?>
```

8.5　帖子分类管理模块设计

8.5.1　帖子分类管理模块概述

本在线论坛是针对不同种类的计算机编程语言开发的一个网络交流平台，所以在本论坛中根据不同种类的计算机编程语言对论坛中的帖子进行分类管理，帖子分类管理模块的运行效果如图 8.18 所示。



图 8.18　帖子分类管理模块的运行效果

8.5.2　使用 SQL 语句查询数据技术

帖子分类管理功能的实现主要应用 mysql_query()、mysql_num_rows()和 mysql_fetch_array()函数，以及 SQL 语句对数据库中的数据进行查询，然后根据不同的语言种类进行分类显示。下面对应用的函数和语句进行详细讲解。

1．mysql_query()函数

返回 MySQL 结果集中一个单元的内容，经常用在获取少量数据的程序中。语法如下：

```
mixed mysql_query ( resource result, int row [, mixed field] )
```

本函数获取一个 query 的结果。参数 field 可以是字段名称、顺序或者是 fieldname.tablename 的格式。如果给列起了别名（“select foo as bar from...”），则可用别名替代列名。

2．mysql_num_rows()函数

使用 mysql_num_rows()函数可以获取由 select 语句查询到的结果集中行的数目。语法如下：

```
int mysql_num_rows ( resource result )
```

此命令仅对 SELECT 语句有效。要取得被 insert、update 或者 delete 语句所影响到的行的数目，再使用 mysql_affected_rows()函数。

3．mysql_fetch_array()函数

返回根据从结果集获取的行生成的数组，如果没有更多行则返回 false。该函数在获取数据库中数据时是非常实用的。语法如下：

```
array mysql_fetch_array ( resource result [, int result_type] )
```

mysql_fetch_array()函数的参数说明如表 8.4 所示。

表 8.4　mysql_fetch_array()函数的参数说明

参　　数	说　　明
result	资源类型的参数，要传入的是由 mysql_query()函数返回的数据指针
result_type	可选项，整数型参数，要传入的是 MYSQL_ASSOC、MYSQL_NUM、MYSQL_BOTH 3 种由 PHP 定义好的常数之一，默认值是 MYSQL_BOTH 用 MYSQL_ASSOC 只得到关联索引（相当于 mysql_fetch_assoc()函数） 用 MYSQL_NUM 只得到数字索引（相当于 mysql_fetch_row()函数） 用 MYSQL_BOTH 将得到一个同时包含关联和数字索引的数组



本函数返回的字段名是区分大小写的。

8.5.3　帖子分类管理模块的实现过程

1．页面设计

在页面设计中，使用图片固然可以使页面变得非常漂亮，但是也存在一些弊端，如果在网页中输出大量的数据时使用图片作为背景，一旦处理不好，会导致数据显示与背景不协调，达不到最佳效果。在网站中使用简单的表格来完成页面的设计就不会出现上述问题，特别是对大量数据输出处理上，使用表格和添加背景色来完成页面设计是非常不错的方法，其中细线表格边线的使用更是程序员设计网页的首选参数。这里帖子分类管理模块就是使用细线表格来完成的，效果如图 8.19 所示。

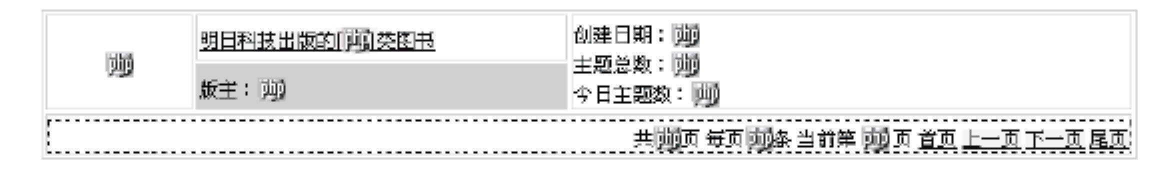


图 8.19　帖子分类管理模块的效果

帖子分类管理模块的页面设计流程如下所示。

（1）使用 include()语句包含数据库服务器连接文件、论坛的头文件、用户登录模块和输出登录成功用户的用户名文件。代码如下：

```
<?php
include("conn/conn.php");
include("index_01.php");
include("index_02.php");
include("index_03.php");
?>
```

（2）设计一个细线表格，对数据库中的数据进行循环输出，这里按照明日科技的不同类别的图书进行帖子的分类管理，其中包括帖子的所属专区、表情图和版主、创建日期、专区帖子的主题总数和今日发布帖子的总数。

细线表格的设计方法是：首先创建一个表格，然后设置表格的边框、填充和间距都为 1，接着设置表格的边框颜色为白色，背景色为红色，最后将表格中单元格的背景色设置为白色，保存表格。这时再浏览这个表格时，就是一个边线为红色的细线表格。

2．代码设计

在本论坛中，对帖子进行分类管理主要应用 while 循环语句和 SQL 语句，循环读取数据库中存储的有关不同语言书籍的数据，并对数据进行分页显示。其关键代码如下：

（代码位置：光盘\TM\Instance\08\index_04.php）

```
<?php
include("conn/conn.php");
if(isset($_GET['page'])){
```



```
            $page=$_GET['page'];
        }else{
            $page=1;
        }
          $page_count=3;
          $select=mysql_query("select * from tb_category",$conn);
          $row=mysql_num_rows($select);
          $page_page=ceil($row/$page_count);
          $offect=($page-1)*$page_count;          //获取上一页的最后一条记录，从而计算下一页的起始记录
          $selects=mysql_query("select * from tb_category where id order by id desc limit $offect,$page_count",
$conn);
          while($array=mysql_fetch_array($selects)){
          $icon=substr($array['icon'],3,30);
?>
<table width="987" height="88" border="1" align="center" bordercolor="#FFCC99">
  <tr>
    <td width="172" rowspan="2" align="center"><?php echo "<img src=\"$icon\">";?></td>
    <td width="4113" height="28"><a href="lb.php?category=<?php echo $array['category'];?>">明日科技出版
的[<?php echo $array['category'];?>]类图书</a></td>
    <td width="340" rowspan="2">创建日期：<?php echo $array['create_date'];?><br>
    主题总数：<?php $selectes=mysql_query("select * from tb_content where category='".$array['category']."'",$conn);
            $count=mysql_num_rows($selectes);
            echo $count;
            ?><br>
    今日主题数：<?php $dates=date("Y-m-d");
            $rows=mysql_query("select * from tb_content where release_date='$dates' and category='".$array
['category']."'",$conn);
            $counts=mysql_num_rows($rows);
            echo $counts;?></td>
  </tr>
  <tr>
    <td height="28" bgcolor="#FFCC66">版主：<?php echo $array['noderator']?></td>
  </tr>
  <?php
}
?>
  <tr>
    <td colspan="3"><div align="center">
      <div align="right">共<?php echo $page_page;?>页 每页<?php echo $page_count;?>条 当前第<?php
echo $page; ?>页
          <a href="index.php?page=1">首页</a>
          <a href="index.php?page=<?php if($page==1){echo $page=1; }else{ echo $page-1; }?>">上一页</a>
          <a href="index.php?page=<?php if($page<$page_page){echo $page+1;}else{ echo $page_page;}?>">
下一页</a>
    <a href="index.php?page=<?php echo $page_page; ?>">尾页</a></div></td>
  </tr>
</table>
```

当单击图书类别超链接时，将显示该类别的所有帖子信息，并进行分页显示。其关键代码如下：

（代码位置：光盘\TM\Instance\08\ lb.php）

```
<?php
include("conn/conn.php");
include("index_01.php");
include("index_02.php");
include("index_03.php");
?>
<?php
if(isset($_GET['page'])){
        $page=$_GET['page'];
}else{
```

```
        $page=1;
}
$page_count=3;
$select=mysql_query("select * from tb_content",$conn);
$row=mysql_num_rows($select);
$page_page=ceil($row/$page_count);
$offect=($page-1)*$page_count;              //获取上一页的最后一条记录，从而计算下一页的起始记录
?>
<table width="96%" border="1" align="center" cellpadding="1" cellspacing="1" bordercolor="#FFFFFF"
bgcolor="#E1DAEA"><tbody>
        <tr>
        <td height="30" colspan="2" align="center" bgcolor="#FFFBF0" class="STYLE4">
<?php
        if(isset($_GET['category'])){
                $category=$_GET['category'];
                echo urlencode($category);
        }else{
                $category="";
        }
?>
                类图书</td>
                <td width="2611" bgcolor="#FFFBF0"> </td>
        </tr></tbody>
</table>
……//省略了部分代码
```

8.6 发帖模块设计

视频讲解：光盘\TM\Video\第 8 章\发帖模块设计.exe

8.6.1 发帖模块概述

发布帖子是论坛中必不可少的功能，用户登录后，单击“发布主题”超链接即可进入发帖模块，发帖模块的运行效果如图 8.20 所示。左侧显示发帖人的一些基本信息，右侧为发帖区域，选择帖子类别，输入主题、内容，选择表情后，单击“主题提交”按钮即可发布帖子。

8.6.2 while 循环语句技术

在处理发帖模块中使用的表情图时，应用 while 语句循环输出表情图片，在读取数据库中存储的二进制格式图像时，应用了 mysql_query()和 mysql_fetch_array()函数。使用$_SESSION 变量输出当前用户信息。

使用 while 语句循环输出表情图片的代码如下：

```
<?php
$select1=mysql_query("select * from tb_expression",$conn);
while($array1=mysql_fetch_array($select1)){
?>
<input type="radio" name="tx" value="<?php echo $array1['id'];?>" /> <img src="<?php echo "image_1.php?
recid=".$array1['id'];?>" width="24" height="24">
<?php
}
?>
```

8.6.3　发帖模块的实现过程

1. 页面设计

发帖模块的设计可以分为两部分，第一部分是输出登录用户的个人信息，第二部分是发帖表单内容的设计。发帖模块的整体设计效果如图 8.21 所示。



图 8.20　发帖模块的运行效果

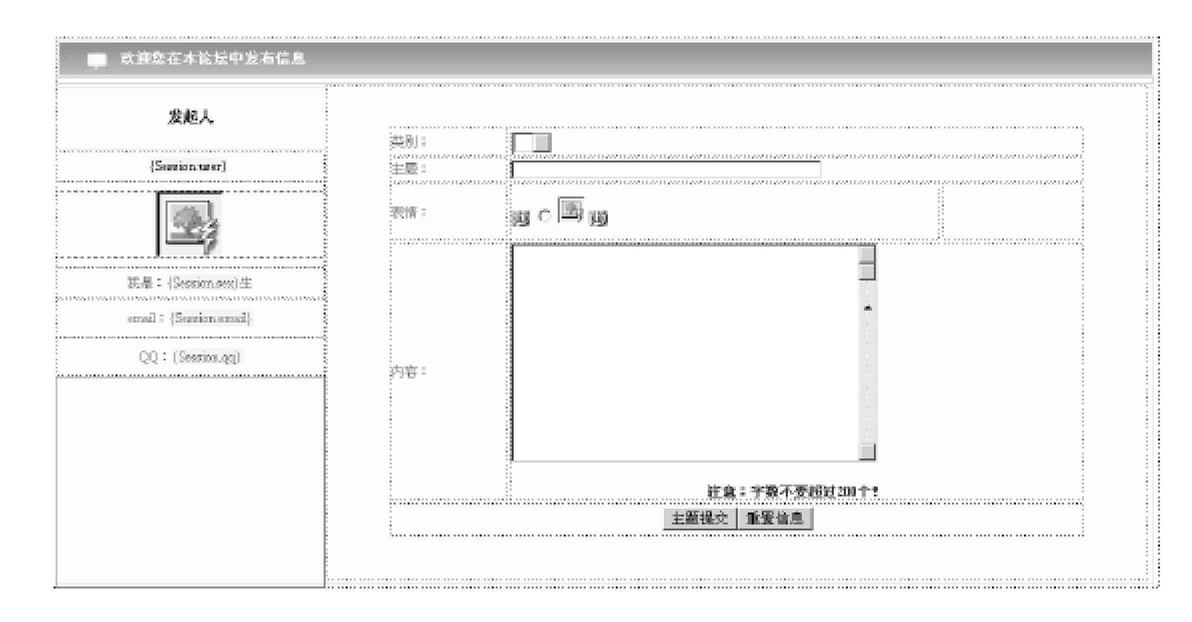


图 8.21　发帖模块的整体设计效果

发帖模块的整体设计流程如下。

（1）使用 include()语句获取论坛的头文件和登录用户的个人信息。代码如下：

```
<?php
include("index_01.php");
include("index_02.php");
?>
```

（2）设计发帖模块中的表单元素。

发帖页面涉及到的 HTML 表单元素如表 8.5 所示。

表 8.5　发帖页面涉及到的 HTML 表单元素

名　　称	类　　型	含　　义	重 要 属 性
form 1	form	表单	method="post" action="fbzt_ok.php" enctype="multipart/form-data" onSubmit="javascript:return fbzt_check();">
category	text	图书类别	<?php echo $array['category']?>"><?php echo $array['category'];?>
subject	text	帖子的主题名称	
tx	radio	表情图	value="<?php echo $array1['id'];?>"
content	text	帖子的内容	
Submit	submit	主题提交	
Submit2	reset	重置信息	

2. 代码设计

发帖模块主要通过一个文件来实现，表单提交及数据的处理文件是 fbzt.php 文件。

（1）在表单提交页 fbzt.php 中，输出登录用户的个人信息和提交的表单，以及论坛中使用的表情图，并且将表单中的数据提交到 fbzt.php 文件中，存储到数据库中。

（2）在 fbzt.php 文件中，将表单提交的数据存储到数据库中。其关键代码如下：

（代码位置：光盘\TM\Instance\08\fbzt.php）

```
<?php
include("conn/conn.php");
$select=mysql_query("select * from tb_category",$conn);
```

```
if(isset($_SESSION['user']) and $_SESSION['user']!=null){
        if(isset($_POST['Submit']) and $_POST['Submit']=="主题提交"){
                $date=date("Y-m-d");
                $insert=mysql_query("insert into tb_content(category,subject,content,username,release_date) values
('".$_POST['category']."','".$_POST['subject']."','".$_POST['content']."','".$_SESSION['user']."','$date')",$conn);
                if($insert){
                        echo "<script>alert('发布成功！');window.location.href='index.php';</script>;";
                }else{
                        echo "<script>alert('发布失败！');window.location.href='fbzt.php';</script>;";
                }
        }
}else{
        echo "<script>alert('请先登录！');window.location.href='index.php';</script>";
}
?>
```

8.7 回帖模块设计

视频讲解：光盘\TM\Video\第 8 章\回帖模块设计.exe

8.7.1 回帖模块概述

回帖模块主要实现的是对用户提出的问题进行回复，无论是版主还是普通用户都可以对提出的问题发表自己的看法和见解。回帖模块的运行效果如图 8.22 所示。

8.7.2 mysql 函数处理技术

其中应用到的 MySQL 函数包括 mysql_query()和 mysql_fetch_array()，与发帖模块和用户注册模块中相同，这里不再赘述。

8.7.3 回帖模块的实现过程

1．页面设计

回帖模块的设计和发帖模块的设计相似，该模块由 4 个大的部分组成，第 1 部分是输出论坛的头文件和登录者的信息，第 2 部分是输出要回复的帖子的发起人信息，第 3 部分是回帖表单的设计，第 4 部分是尾文件的设计。回帖模块的设计效果如图 8.23 所示。



图 8.22　回帖模块的运行效果

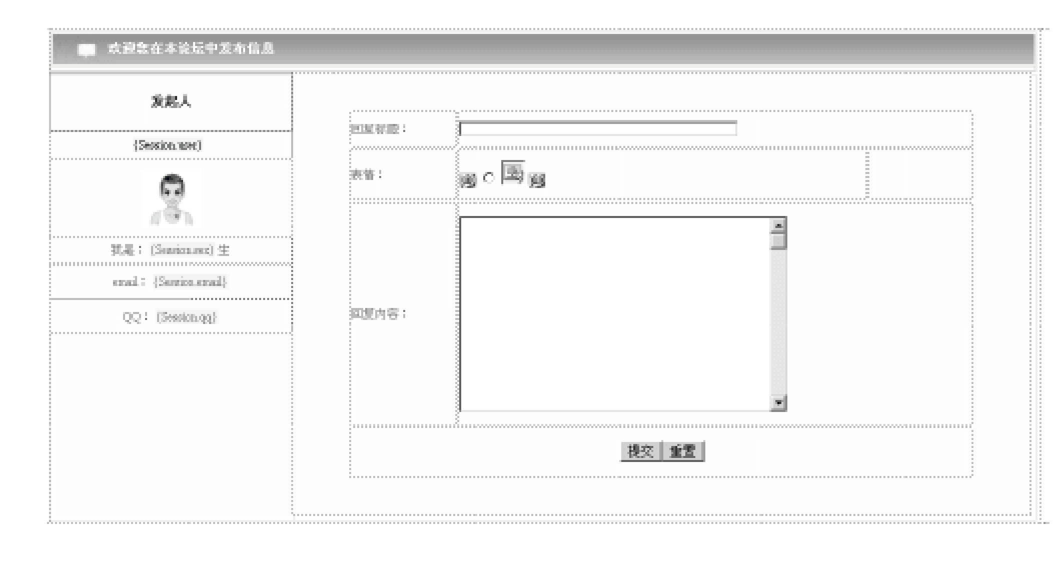


图 8.23　回帖模块的设计效果

回帖模块的整体设计流程如下。

（1）使用 include()语句获取论坛的头文件和用户登录信息。代码如下：

```
<?php
include("conn/conn.php");
include("index_01.php");
include("index_02.php");
?>
```

（2）输出帖子发起人的信息，包括用户名、图片、QQ 和邮箱等，并且输出发布帖子的主题。

（3）设计回复帖子提交的表单元素，包括主题、回复内容和一些隐藏的选项。

回帖页面涉及到的 HTML 表单元素如表 8.6 所示。

表 8.6　回帖页面涉及到的 HTML 表单元素

名　　称	类　　型	含　　义	重 要 属 性
Form1	form	表单	method="post" action="fbzt_ok.php?subject=<?php echo $subject;?>" enctype="multipart/form-data"
subject	text	回复主题	
content	textarea	回复内容	
Submit		提交	
Submit2		重置	

2．代码设计

回帖功能的实现也通过一个文件（hfzt.php 文件）来完成，该文件完成帖子回复表单的设计及对表单提交的内容进行处理。hfzt.php 文件的关键代码如下：

（代码位置：光盘\TM\Instance\08\hfzt.php）

```
<?php
session_start();                                        //初始化 SESSION 变量
if(isset($_SESSION['user'])){                           //判断是否是会员登录
include("conn/conn.php");                               //连接数据库
include("index_01.php");
include("index_02.php");
if(isset($_POST['subject']) and $_POST['Submit']=="提交"){  //判断提交数据是否存在
    $select=mysql_query("select * from tb_content where id='".$_GET['h_id']."'",$conn);
    $array=mysql_fetch_array($select);
    $category=$array['category'];
    $subject=$array['subject'];
    $date=date("Y-m-d");
    $insert=mysql_query("insert  into  tb_resume_contents(resume_subject,resume_contents,resume_date,
username,category,subject) values('".$_POST['subject']."','".$_POST['content']."','$date','".$_SESSION['user']."',
'$category','$subject')",$conn);
    if($insert){
        echo "<script>alert('回复成功！');window.location.href='lb.php';</script>";
    }else{
        echo "<script>alert('回复失败！');window.location.href='lb.php';</script>";
    }
}
?>
……//省略了部分代码
<?php
    }else{
        echo "<script>alert('您没有登录');window.location.href='index.php';</script>";
}
?>
```

8.8 后台管理模块设计

视频讲解：光盘\TM\Video\第 8 章\后台管理模块设计.exe

8.8.1 后台管理模块概述

在后台管理模块中，主要实现对论坛中注册用户、发布的帖子、回复帖子和非法关键字进行管理。后台管理模块主页运行效果如图 8.24 所示。

8.8.2 URL 编码和 SWITCH 框架技术

后台管理模块中主要应用的函数基本已经在前台的内容中详细介绍过了，这里介绍前面没有提到的 urlencode()函数和 switch 语句。

1. urlencode()函数

本函数对指定的中文字符串进行编码，返回值为字符串类型。urlencode()函数将传入的字符串进行编码。其语法如下：

```
string urlencode ( string str )
```

参数 str 为指定要编码的字符串。

注意 使用 GET 方法传递网址时，如果不使用编码方式则可能会发生获取的数据信息不完整的现象，使用 urlencode()函数进行编码可避免该问题的发生，而且又能保证传递数据的安全性，可谓是一举两得。

2. switch 语句

通过 switch 语句可以实现一个简单的框架网页的效果，这种框架网页的效果在论坛的后台管理中是非常实用的。switch 语句的语法如下：

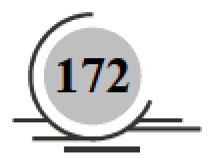
```
switch ( expr ){                                    //expr 条件为变量名称
        case expr1:                                 //case 后的 expr1 为变量的值
              statement1;                           //符合该条件时要执行的部分
           break ;                                  //应用 break 来跳离循环体
        case expr2 :
              statement2 ;
           break ;
       default:
           statementN;
           break;
}
```

参数 expr 是表达式的值，即 switch 语句的条件变量的名称；参数 expr1 放置于 case 语句之后，是要与条件变量 expr 进行匹配的值中的一个；statement1 是在参数 expr1 的值与条件变量 expr 的值相匹配时执行的代码；break 语句实现终止语句的执行，即语句在执行过程中，遇到 break 就停止执行，跳出循环体；default 是 case 的一个特例，匹配了任何其他 case 都不匹配的情况，并且是最后一条 case 语句。

8.8.3 后台主页的实现过程

1. 页面设计

后台管理模块主页应用的是简单的框架效果，通过 switch 语句来完成，可以分为 3 个部分。后台管理



模块的设计效果如图 8.25 所示。



图 8.24　后台管理系统主页的运行效果

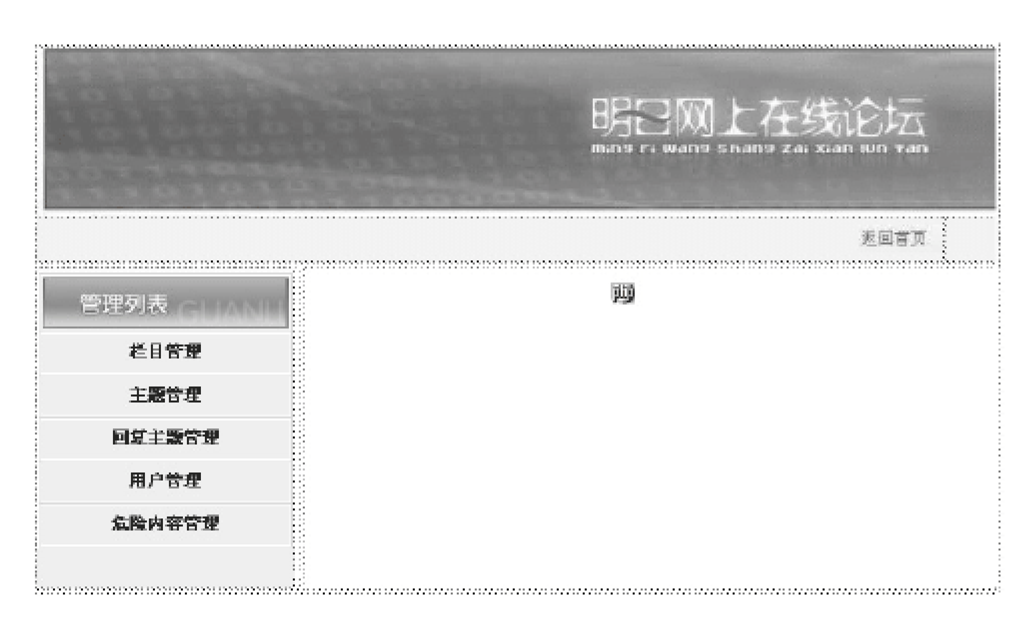


图 8.25　后台管理模块主页的设计效果

后台管理模块主页的整体设计流程如下。

（1）使用 include()语句获取论坛的头文件。代码如下：

```
<?php include("index_01.php");?>
```

（2）设计后台管理中的栏目选项，由 5 个栏目组成。

（3）根据单击不同栏目的超链接输出对应栏目的内容，默认输出栏目管理的内容。程序代码如下：

```
<?php switch($pt){
        case "栏目管理":
            include "lmgl.php";
        break;
        case "主题管理":
            include "ztgl.php";
        break;
        case "回复主题管理":
            include "hfztgl.php";
        break;
        case "用户管理":
            include "hygl.php";
        break;
        case "危险内容管理":
            include "ss.php";
        break;
        default :
            include "lmgl.php";
        break;
        }
?>
```

2．代码设计

后台主页 index.php 文件，通过 switch 语句创建网页框架，实现在不同功能模块之间的跳转操作。其关键代码如下：

（代码位置：光盘\TM\Instance\08\admin\index.php）

```
    <tr>
        <td>
            <a href="index.php?lmbs=<?php echo urlencode("栏目管理");?>"><img src="images/02_03.gif"
alt="" width="1711" height="211" border="0"></a></td>
    </tr>
    <tr>
        <td>
```

```
                <a href="index.php?lmbs=<?php echo urlencode("主题管理");?>"><img src="images/02_06.gif" alt="" width="1711" height="23" border="0"></a></td>
    </tr>
    <tr>
        <td>
            <a href="index.php?lmbs=<?php echo urlencode("回复主题管理");?>"><img src="images/02_08.gif" alt="" width="1711" height="23" border="0"></a></td>
    </tr>
    <tr>
        <td>
            <a href="index.php?lmbs=<?php echo urlencode("用户管理");?>"><img src="images/02_10.gif" alt="" width="1711" height="24" border="0"></a></td>
    </tr>
    <tr>
        <td>
            <a href="index.php?lmbs=<?php echo urlencode("危险内容管理");?>"><img src="images/02_12.gif" alt="" width="1711" height="23" border="0"></a></td>
    </tr>
……//省略了部分代码
```

8.8.4　栏目管理模块的实现过程

在后台管理系统中主要实现 5 个功能：栏目管理、主题管理、回复主题管理、用户管理和危险内容管理。功能实现的方法基本都是相同的，这里以栏目管理为例进行讲解。其运行效果如图 8.26 所示。



图 8.26　栏目管理模块的运行效果

栏目管理模块由 lmgl.php 和 delete3.php 两个文件组成。在 lmgl.php 文件中完成栏目管理的大部分操作，可以将其分解为 3 部分内容。

（1）创建栏目添加的表单，完成栏目的添加操作，并且将数据提交到本页，在本页中将数据添加到指定的数据表中。其关键代码如下：

```
<form action="index.php?lmbs=栏目管理" method="post" enctype="multipart/form-data" name="myform" id="myform">
    <tr>
        <td width="170" height="30" align="middle" bgcolor="#FFFBF0" class="style1"> 版主:
          <input id="noderator" size="111" name="noderator" />
        </td>
        <td width="200" align="middle" bgcolor="#FFFBF0" class="style1">所属专区:
          <select id="category" size="1" name="category">
              <option value="asp" selected="selected">ASP</option>
              <option value="jsp">JSP</option>
              <option value="delphi">Delphi</option>
              <option value="visual basic">Visual Basic</option>
              <option value="visual foxpro">Visual Foxpro</option>
              <option value="visual c++">Visual C++</option>
              <option value="power">Power Buider</option>
              <option value=".net">.net</option>
          </select></td>
        <td align="left" valign="center" bgcolor="#FFFBF0" class="style1">  图标:
```

```
                <input type="radio" name="icon" value="<?php echo $top;?>" />
              <img src='../images/tx/photo.jpg' width='411' height='40' />
              <input type="radio" name="icon" value="<?php echo $df;?>" />
              <img src='../images/tx/photoes.jpg' width='411' height='40' /></td>
        </tr>
        <tr>
          <td colspan="2" bgcolor="#FFFBF0"> </td>
          <td bgcolor="#FFFBF0" class="style1">  <input id="zhuijia" type="submit" value="追加栏目"
name="zhuijia" /></td>
        </tr>
</form>
<?php
include("../conn/conn.php");
$top=("../images/tx/photo.jpg");
$df=("../images/tx/photoes.jpg");
if(isset($_POST['noderator']) and $_POST['zhuijia']=="追加栏目"){
     $date=date("Y-m-d");
     $insert=mysql_query("insert into tb_category(icon,category,noderator,create_date) values('".$_POST['icon']."',
'".$_POST['category']."','".$_POST['noderator']."','$date')",$conn);
     if($insert){
          echo "<script>alert('追加成功！');window.location.href='index.php?lmbs=".$_GET['lmbs']."';</script>";
     }else{
          echo "<script>alert('追加失败！');window.location.href='index.php?lmbs=".$_GET['lmbs']."';</script>";
     }
}
?>
```

（2）执行查询语句，通过 while 语句完成数据库中存储栏目数据的分页输出。其关键代码如下：

```
<?php
     if(isset($_GET['page'])){
          $page=$_GET['page'];
     }else{
          $page=1;
     }
       $page_count=3;
       $select=mysql_query("select * from tb_category",$conn);
       $row=mysql_num_rows($select);
       $page_page=ceil($row/$page_count);
       $offect=($page-1)*$page_count;          //获取上一页的最后一条记录，从而计算下一页的起始记录
       $selects=mysql_query("select * from tb_category where id order by id desc limit $offect,$page_count",$conn);
      if($selects){
        while($myrow=mysql_fetch_array($selects)){   ?>
    <tr class="style1" align="middle">
      <td height="44" align="center" bgcolor="#FFFBF0"><img src="<?php echo $myrow['icon'];?>" width="40"
height="40" /></td>
      <td height="44" align="center" bgcolor="#FFFBF0"><?php echo $myrow['category'];?></td>
      <td height="44" align="center" bgcolor="#FFFBF0"><?php echo $myrow['noderator'];?></td>
      <td height="44" align="center" bgcolor="#FFFBF0"><?php echo $myrow['create_date'];?></td>
      <td height="44" align="center" bgcolor="#FFFBF0"><a href="delete3.php?lmbs=<?php echo urlencode("
栏目管理");?>&amp;id=<?php echo $myrow['id'];?>">删除</a></td>
    </tr>
         <?php }}?>
```

（3）创建分页超链接，实现不同页面之间的跳转。其关键代码如下：

```
<table width="80%" border="0" cellspacing="0" cellpadding="0">
<tr class="style4">
     <td width="110%" class="#ff0000">  页次：<?php echo $page;?>/<?php echo $page_page;?>
```

```
页 记录：<?php echo $row;?>条  </td>
          <td width="39%" class="#ff0000">
              <a href="index.php?lmbs=<?php echo urlencode($_GET['lmbs']);?>&amp; page=1">首页</a>
              <a href="index.php?lmbs=<?php echo urlencode($_GET['lmbs']);?>&amp; page=<?php if($page==1)
{echo $page=1; }else{ echo $page-1; }?>">上一页</a>
              <a href="index.php?lmbs=<?php echo urlencode($_GET['lmbs']);?>&amp; page=<?php if($page<
$page_page){echo $page+1;}else{ echo $page_page;}?>">下一页</a>
              <a href="index.php?lmbs=<?php echo urlencode($_GET['lmbs']);?>&amp; page=<?php echo
$page_page; ?>">尾页</a></td>
         </tr>
</table>
```

（4）删除栏目信息，根据 lmgl.php 页面传递的链接 ID，在 delete3.php 文件中执行栏目的删除操作。其关键代码如下：

```
<?php
include("../conn/conn.php");
$delete=mysql_query("delete from tb_category where id='".$_GET['id']."'",$conn);
            if($delete){
                echo "<script>alert('删除成功！');window.location.href='index.php?lmbs=".$_GET['lmbs']."';</script>";
     }else{
                echo "<script>alert('删除失败！');window.location.href='index.php?lmbs=".$_GET['lmbs']."';</script>";
}
?>
```

8.9 小　　结

本章主要介绍了在线论坛的基本创建流程和开发思路，以及各个功能模块的实现方法。论坛系统的实现主要把握 3 个方面的内容即可，即如何向数据库中添加数据、如何从数据库中读取数据及如何将数据库中的数据删除。论坛中所有功能模块的实现都是在围绕着这 3 个方面进行，用户注册模块、发帖模块、回帖模块是在向数据库中添加内容，而查看帖子模块、帖子分类管理模块是在读取数据库中的数据。在把握住这 3 个大方向以后就是对细节进行处理，可以根据个人的能力和喜好开发出各式各样的论坛系统。

第 2 篇

初级开发

第 9 章

MySQL 数据库

（视频讲解：25 分钟）

数据库作为程序中数据的主要载体，在整个项目中扮演着重要的角色。PHP 自身可以与大多数数据库进行连接，但 MySQL 数据库是开源界所公认的与 PHP 兼容性最好的数据库。MySQL 具有安全性、跨平台性、体积小和高效等特点，可谓 PHP 的“黄金搭档”。MySQL 操作简便、易学易用，受到很多业界人士的青睐。

通过阅读本章内容，你可以：

- 了解 MySQL 数据库的特点
- 掌握 MySQL5 的特性
- 掌握如何连接与断开 MySQL 数据库
- 掌握如何创建、删除数据库
- 掌握如何创建、修改和删除数据表
- 掌握如何管理数据表
- 掌握 phpMyAdmin 的安装配置

9.1　MySQL 简介

PHP 在开发 Web 站点或一些管理系统时，需要对大量的数据进行保存。XML 文件和文本文件虽然可以作为数据的载体，但不易进行管理和对大量数据的存储，所以在项目开发时，数据库就显得非常重要。PHP 可以连接的数据库种类较多，其中 MySQL 数据库与其兼容较好，在 PHP 数据库开发中被广泛应用。

9.1.1　什么是 MySQL

视频讲解：光盘\TM\Video\第 9 章\什么是 MySQL.exe

MySQL 是一款安全、跨平台、高效的，并与 PHP、Java 等主流编程语言紧密结合的数据库系统。该数据库系统是由瑞典的 MySQL AB 公司开发、发布并支持，由 MySQL 的初始开发人员 David Axmark 和 Michael Monty Widenius 于 1995 年建立的。MySQL 的象征符号是一只名为 Sakila 的海豚，代表着 MySQL 数据库的速度、能力、精确和优秀本质。

目前 MySQL 被广泛应用在 Internet 上的中小型网站中。由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，使得很多公司都采用 MySQL 数据库以降低成本。

MySQL 数据库可以称得上是目前运行速度最快的 SQL 语言数据库之一。除了具有许多其他数据库所不具备的功能外，MySQL 数据库还是一种完全免费的产品，用户可以直接通过网络下载 MySQL 数据库，而不必支付任何费用。

9.1.2　MySQL 的特点

视频讲解：光盘\TM\Video\第 9 章\MySQL 的特点.exe

MySQL 具有以下主要特点。

- ☑ 功能强大：MySQL 中提供了多种数据库存储引擎，其各有所长，适用于不同的应用场合，用户可以选择最合适的引擎以得到最高性能，可以处理每天访问量超过数亿的高强度的搜索 Web 站点。MySQL 5 支持事务、视图、存储过程、触发器等。
- ☑ 支持跨平台：MySQL 支持至少 20 种以上的开发平台，包括 Linux、Windows、FreeBSD、IBMAIX、AIX、FreeBSD 等。这使得在任何平台下编写的程序都可以进行移植，而不需要对程序做任何的修改。
- ☑ 运行速度快：高速是 MySQL 的显著特性。在 MySQL 中，使用了极快的 B 树磁盘表（MyISAM）和索引压缩；通过使用优化的单扫描多连接，能够极快地实现连接；SQL 函数使用高度优化的类库实现，运行速度极快。
- ☑ 支持面向对象：PHP 支持混合编程方式。编程方式可分为纯粹面向对象、纯粹面向过程、面向对象与面向过程混合 3 种方式。
- ☑ 安全性高：灵活和安全的权限与密码系统，允许基本主机的验证。连接到服务器时，所有的密码传输均采用加密形式，从而保证了密码的安全。
- ☑ 成本低：MySQL 数据库是一种完全免费的产品，用户可以直接通过网络下载。
- ☑ 支持各种开发语言：MySQL 为各种流行的程序设计语言提供支持，为它们提供了很多的 API 函数，包括 PHP、ASP.NET、Java、Eiffel、Python、Ruby、Tcl、C、C++、Perl 语言等。
- ☑ 数据库存储容量大：MySQL 数据库的最大有效表尺寸通常是由操作系统对文件大小的限制决定的，而不是由 MySQL 内部限制决定的。InnoDB 存储引擎将 InnoDB 表保存在一个表空间内，该表空间可由数个文件创建，表空间的最大容量为 64TB，可以轻松处理拥有上千万条记录的大型数据库。

- ☑ 支持强大的内置函数：PHP 中提供了大量内置函数，几乎涵盖了 Web 应用开发中的所有功能。它内置了数据库连接、文件上传等功能，MySQL 支持大量的扩展库，如 MySQLi 等，可以为快速开发 Web 应用提供便利。

9.1.3 MySQL 5 支持的特性

MySQL 5 已经是一个非常成熟的企业级应用的数据库管理系统，在许多大型的开源项目中被广泛地应用。MySQL 5 支持许多基本和高级的特性，例如：

- ☑ 支持各种数据类型。
- ☑ 支持事物、主键、外键、行级锁定等特性。
- ☑ select 查询语句和 where 字句中，提供完整的操作符和函数支持。
- ☑ 支持子查询。
- ☑ 支持 group by 和 order by 子句。
- ☑ 支持各种聚合函数。
- ☑ 支持 left outer join 和 right outer join 多表连接查询。
- ☑ 支持表别名、字段别名。
- ☑ 支持跨库多表连接查询。
- ☑ 支持查询缓存，能够极大地提升查询性能。
- ☑ 支持存储过程、视图和触发器等特性。
- ☑ 支持多平台、多 CPU 等特性。
- ☑ 支持嵌入式，可以将 MySQL 集成到嵌入式程序中。

9.2 MySQL 下载

视频讲解：光盘\TM\Video\第 9 章\MySQL 下载.exe

MySQL 是一款开源的数据库软件，由于其免费特性得到了全世界用户的喜爱，是目前使用人数最多的数据库。下面将详细讲解如何下载该数据库。

（1）打开 MySQL 主页，地址是 http://www.mysql.com/，如图 9.1 所示。

（2）单击页面上部的 Downloads (GA)链接，如图 9.2 所示。

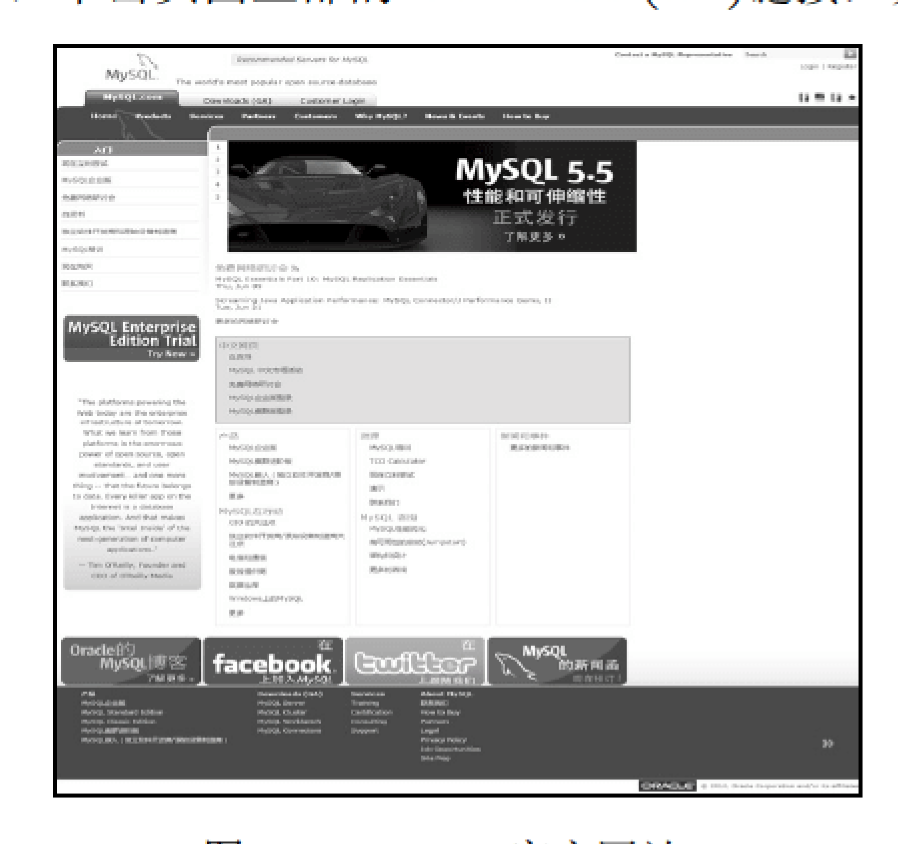
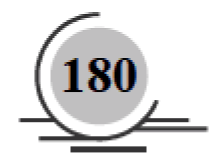


图 9.1　MySQL 官方网站



图 9.2　单击下载页面链接



（3）在新页面中提供了 MySQL、MySQL Workbench、Connector/J（MySQL JDBC 驱动）的下载，如图 9.3 所示。

（4）在图 9.3 中，单击 MySQL Community Server 5.5 链接，在新页面中选择 Windows (x86, 32-bit), MSI Installer 选项，单击 Download 按钮，如图 9.4 所示。

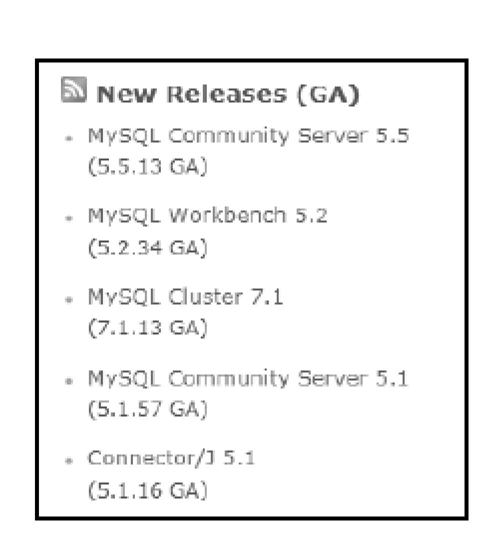


图 9.3　常用的下载链接

Windows (x86, 32-bit), MSI Installer	5.5.13	27.7M	Download
(mysql-5.5.13-win32.msi)	MD5: a05d13d28b67ba1d7dea7b20578132fa | Signature		
Windows (x86, 64-bit), MSI Installer	5.5.13	28.8M	Download
(mysql-5.5.13-winx64.msi)	MD5: 7dad91b677a70050bed7d9adbbc4a51f | Signature		
Windows (x86, 32-bit), ZIP Archive	5.5.13	27.0M	Download
(mysql-5.5.13.zip)	MD5: 4c1fa36cc48960aa1f12304c19e04b68 | Signature		
Windows (x86, 32-bit), ZIP Archive	5.5.13	133.3M	Download
(mysql-5.5.13-win32.zip)	MD5: a84978560037fd757ad529978be80a7b | Signature		
Windows (x86, 64-bit), ZIP Archive	5.5.13	135.5M	Download
(mysql-5.5.13-winx64.zip)	MD5: 823956738f65727919acc3f6e4247049 | Signature		

图 9.4　选择下载的 MySQL 版本

（5）在新页面中单击 No thanks, just take me to the downloads!链接，跳过注册步骤，如图 9.5 所示。

（6）在新页面中选择 Asia 中的一个 HTTP 链接进行下载，如图 9.6 所示。

» No thanks, just take me to the downloads!

图 9.5　直接去下载页面



图 9.6　选择下载的链接

（7）对下载的文件，选择执行操作如图 9.7 所示。

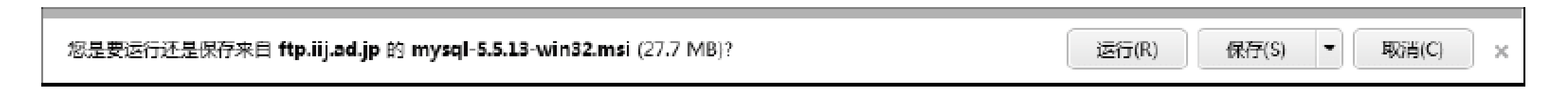


图 9.7　选择执行的操作

（8）下载进度如图 9.8 所示。

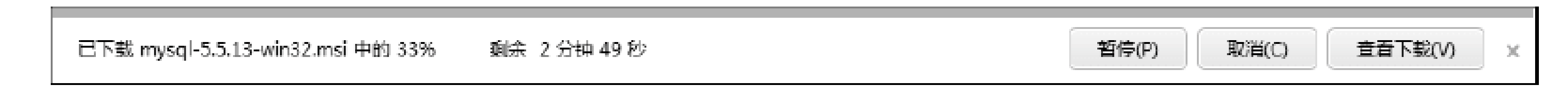


图 9.8　下载进度提示

（9）下载完成如图 9.9 所示。



图 9.9　下载完成提示



虽然 IE9 浏览器提示该文件可能有危险，可以单击右侧的×号关闭这个提示信息。

9.3　MySQL 的环境安装

MySQL 安装文件 mysql-5.5.13-win32.msi 下载完成后，下面讲解其安装过程。

（1）双击运行下载后的程序，如图 9.10 所示。

（2）在图 9.10 中单击“运行”按钮，显示如图 9.11 所示的对话框。



图 9.10　询问是否打开文件

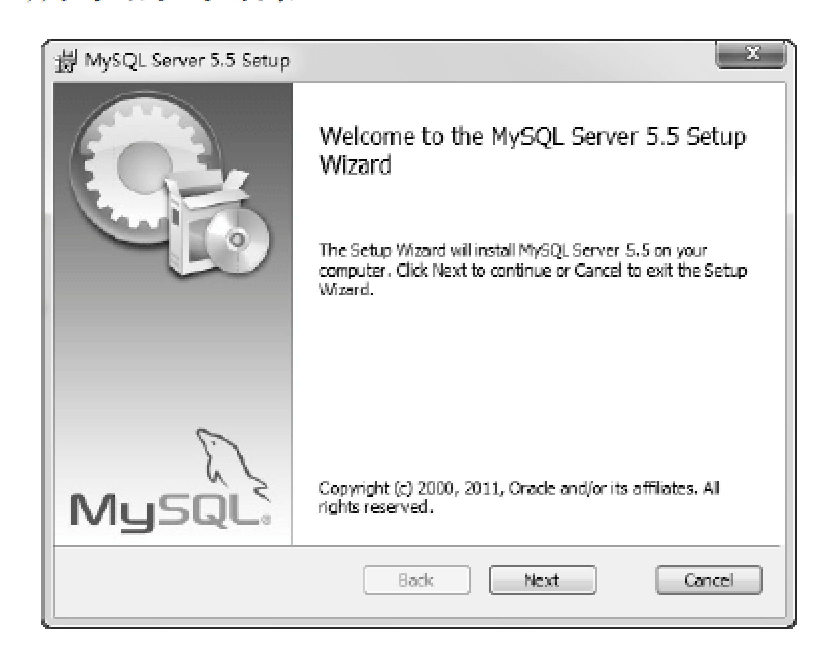


图 9.11　开始运行 MySQL 安装向导

（3）在图 9.11 中单击 Next 按钮，显示如图 9.12 所示的对话框。

（4）在图 9.12 中单击 Next 按钮，将弹出如图 9.13 所示的安装类型选择对话框。

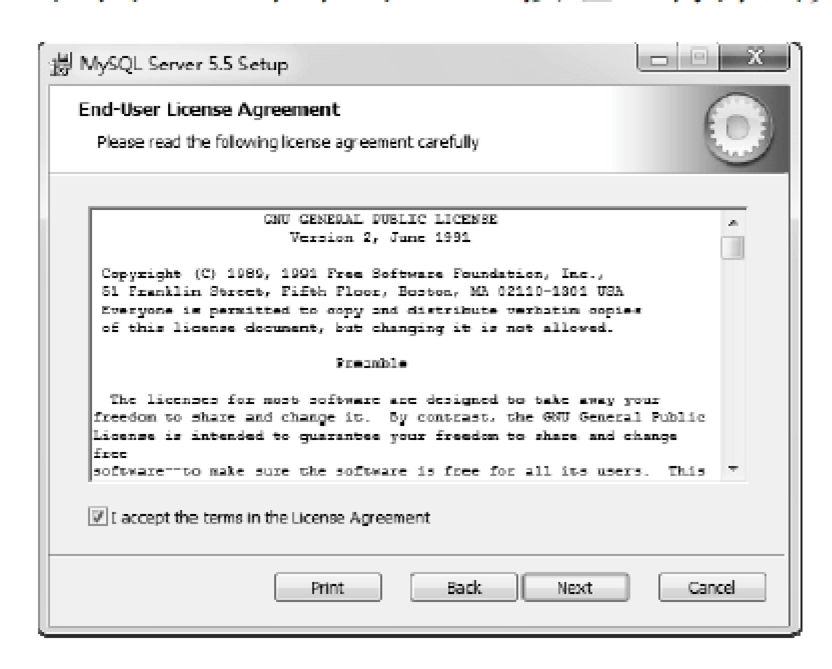


图 9.12　询问是否接受协议

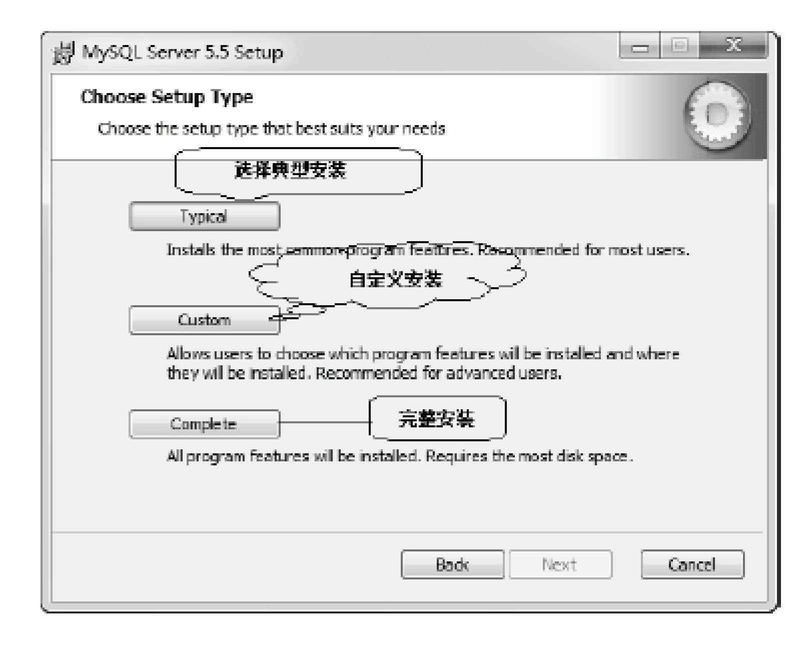


图 9.13　选择安装类型

（5）在图 9.13 中单击 Typical 按钮，显示如图 9.14 所示的对话框。

（6）在图 9.14 中单击 Install 按钮开始安装，安装进度如图 9.15 所示。

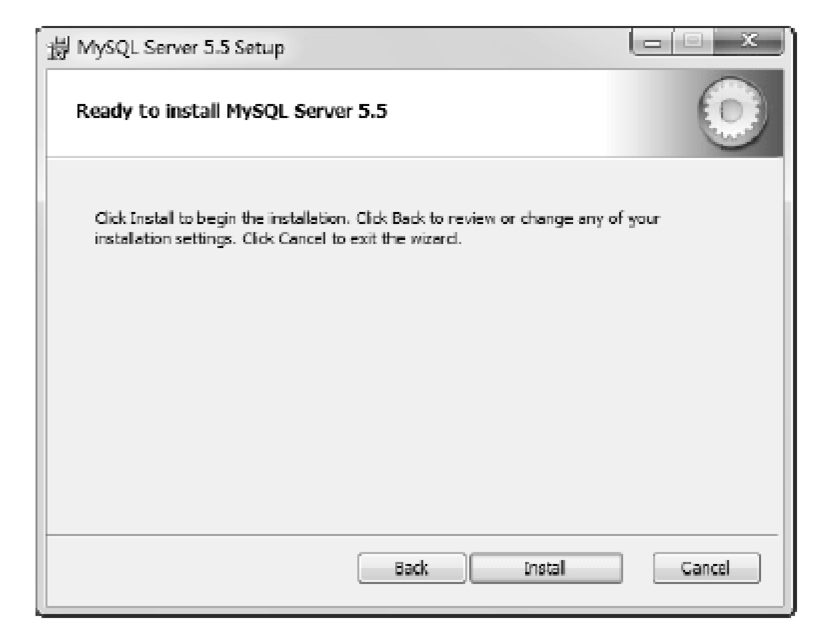


图 9.14　确认前面各选择步骤的对话框

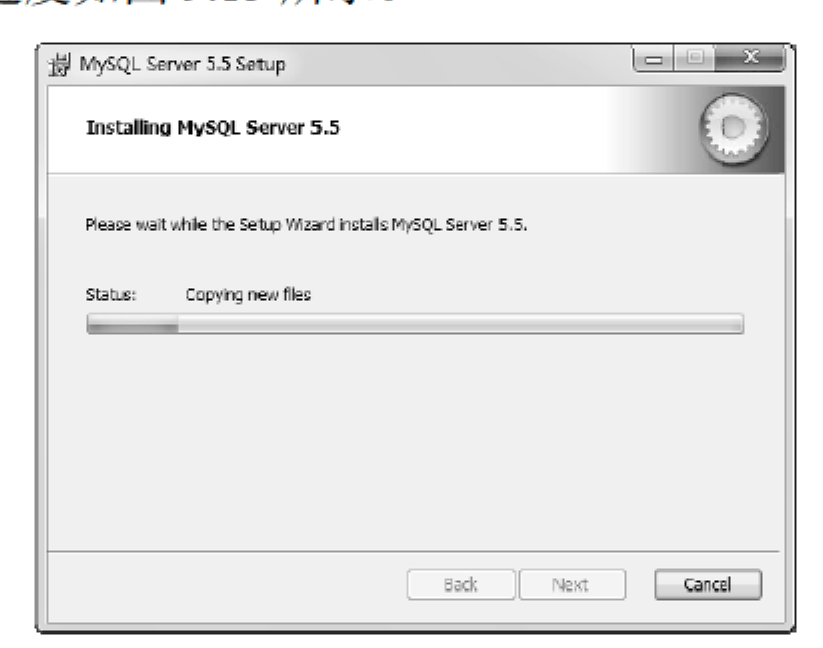
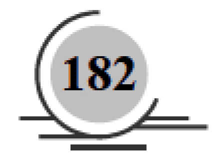


图 9.15　MySQL 安装进度对话框



（7）在安装完成后会显示如图 9.16 和图 9.17 两个广告对话框。

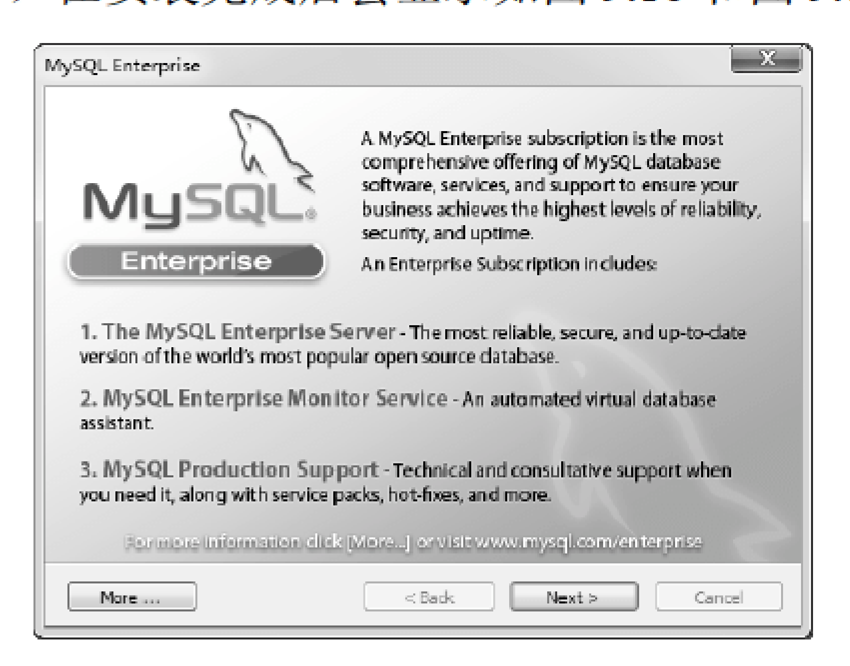


图 9.16　广告对话框 1



图 9.17　广告对话框 2

（8）在图 9.17 中，单击 Next 按钮，显示如图 9.18 所示的对话框。

（9）在图 9.18 中，选中 Launch the MySQL Instance Configuration Wizard 复选框，单击 Finish 按钮，显示如图 9.19 所示的对话框。



图 9.18　安装完成对话框



图 9.19　开始对 MySQL 数据库进行配置

（10）在图 9.19 中，单击 Next 按钮，显示如图 9.20 所示的对话框。

（11）在图 9.20 中，选中 Detailed Configuration（详细配置）单选按钮，单击 Next 按钮，显示如图 9.21 所示的对话框。

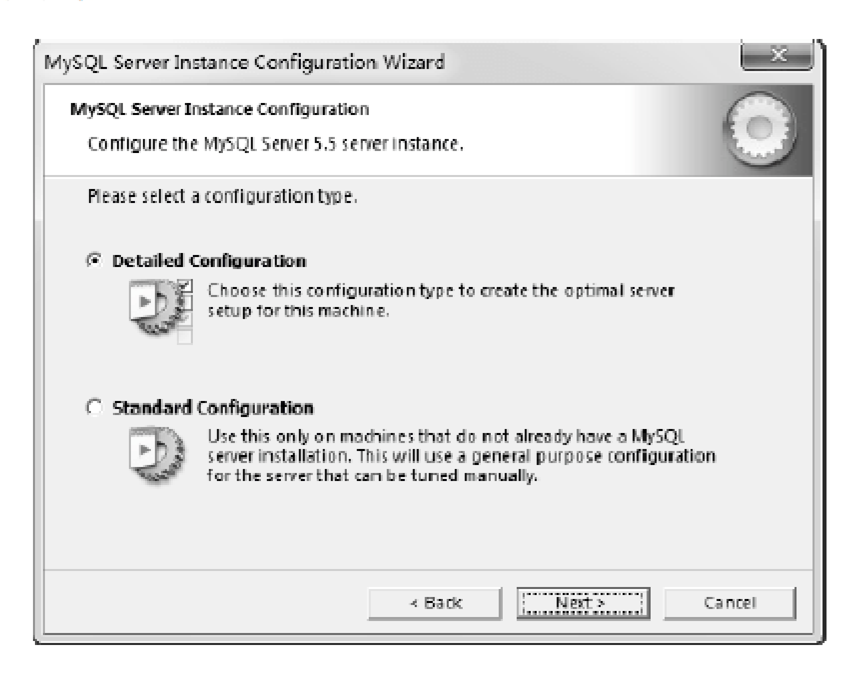


图 9.20　选择使用哪种配置方式

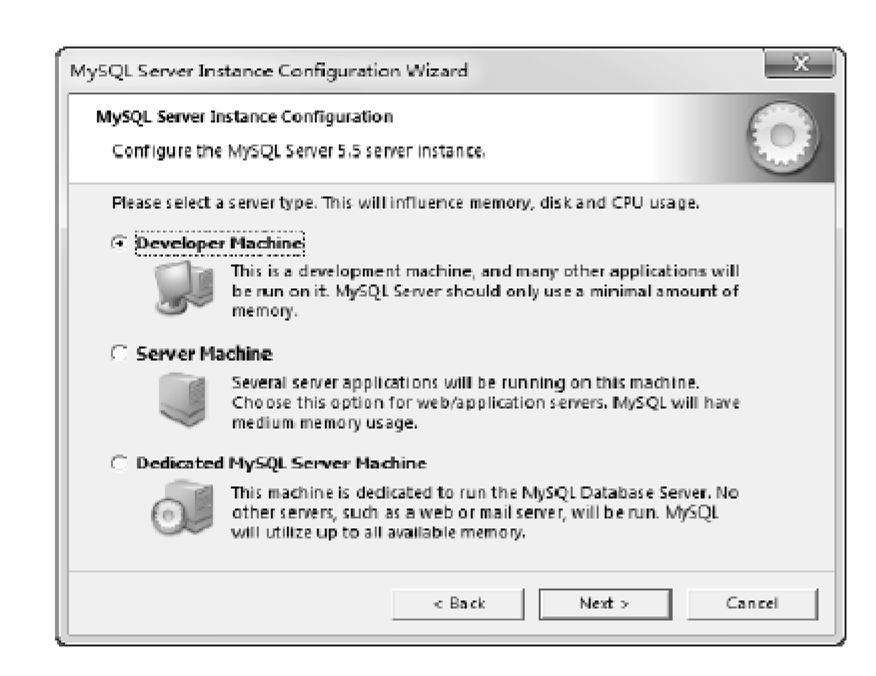


图 9.21　选择服务器类型

（12）在图 9.21 中，选中 Developer Machine（开发者机器）单选按钮，单击 Next 按钮，如图 9.22 所示。

（13）在图 9.22 中，选中 Multifunction Database（多功能数据库）单选按钮，单击 Next 按钮，显示如

图 9.23 所示的对话框。

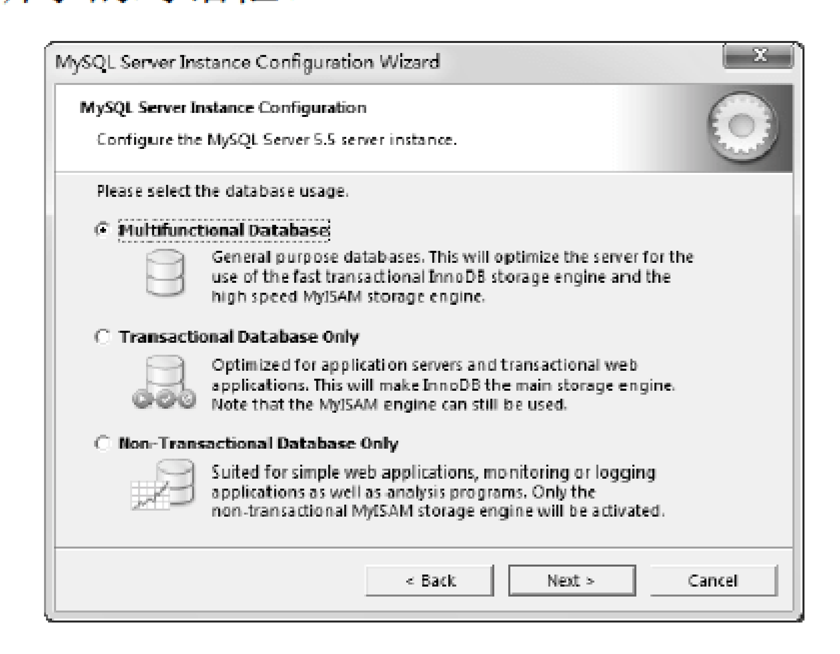


图 9.22　选择数据库类型

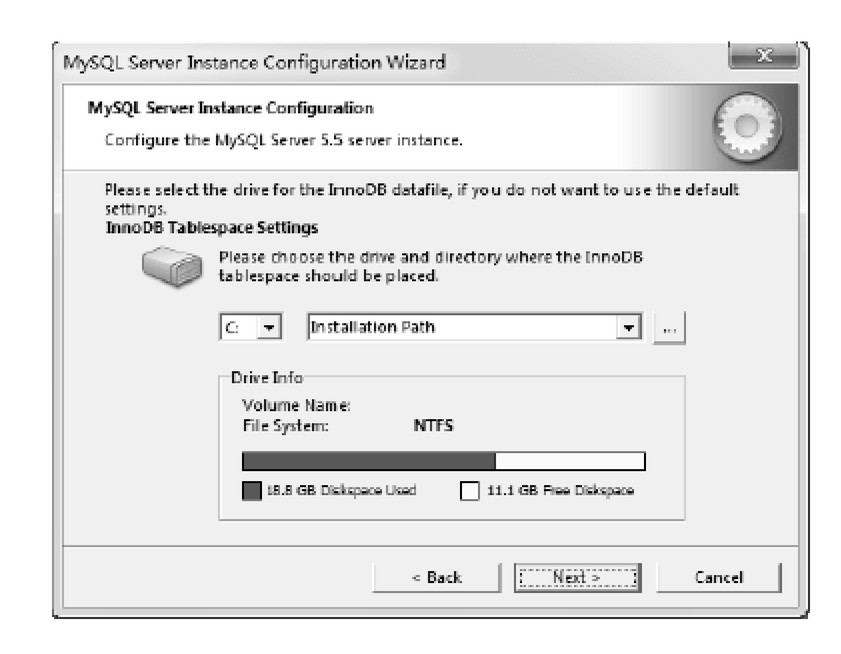


图 9.23　选择 InnoDB 表空间保存位置

（14）在图 9.23 中，使用默认设置，单击 Next 按钮，显示如图 9.24 所示的对话框。

（15）在图 9.24 中，使用默认设置，单击 Next 按钮，显示如图 9.25 所示的对话框。

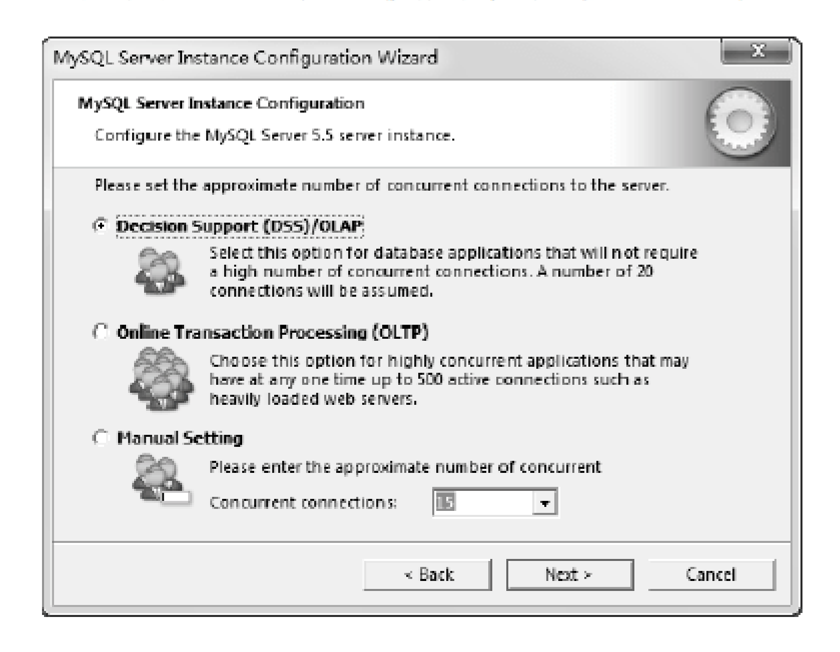


图 9.24　选择服务器并发访问人数

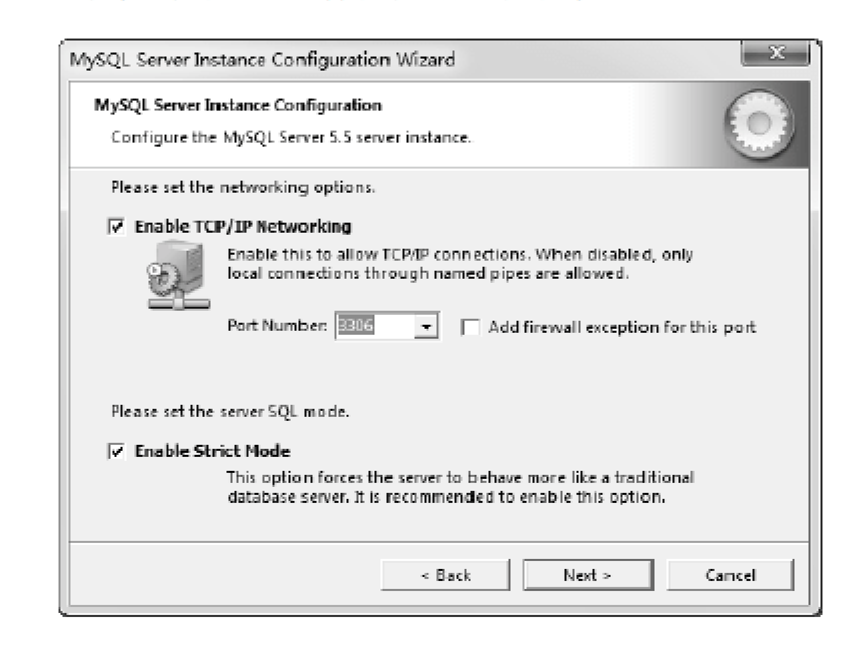


图 9.25　设置端口号和服务器 SQL 模式

注意 MySQL 使用的默认端口是 3306，在安装时，可以修改为其他的，如 3307。但是一般情况下，不要修改默认的端口号，除非 3306 端口已经被占用。

（16）在图 9.25 中，使用默认设置，单击 Next 按钮，显示如图 9.26 所示的对话框。

（17）在图 9.26 中，选中 Manual Selected Default Character Set / Collation 单选按钮，设置字符集编码为 utf8，单击 Next 按钮，显示如图 9.27 所示的对话框。

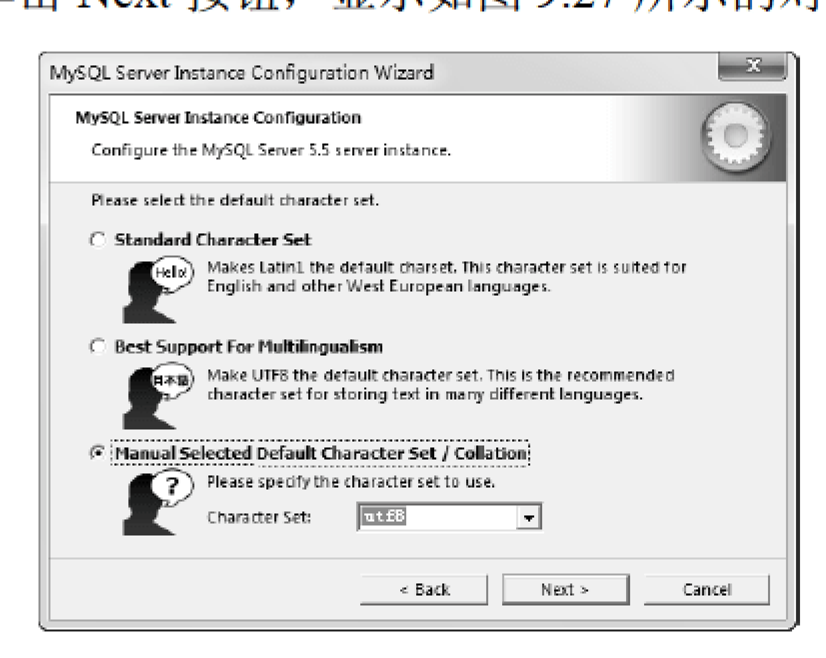


图 9.26　设置默认的字符集

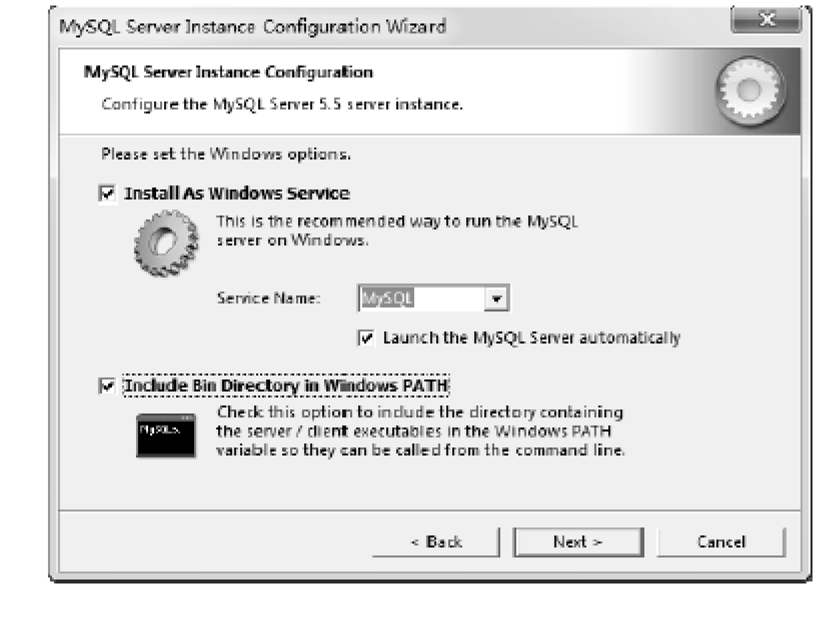
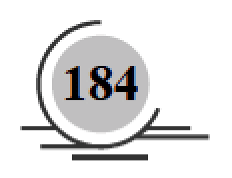


图 9.27　针对 Windows 系统进行的设置

（18）在图 9.27 中，选中 Install As Windows Service 和 Include Bin Directory in Windows PATH 复选框，单击 Next 按钮，显示如图 9.28 所示的对话框。



注意 在安装 MySQL 数据库时，一定要牢记在上述步骤中设置的默认用户 root 的密码，这是我们在访问 MySQL 数据库时必须使用的。

（19）在图 9.28 中输入数据库的密码“111”，单击 Next 按钮，显示如图 9.29 所示的对话框。

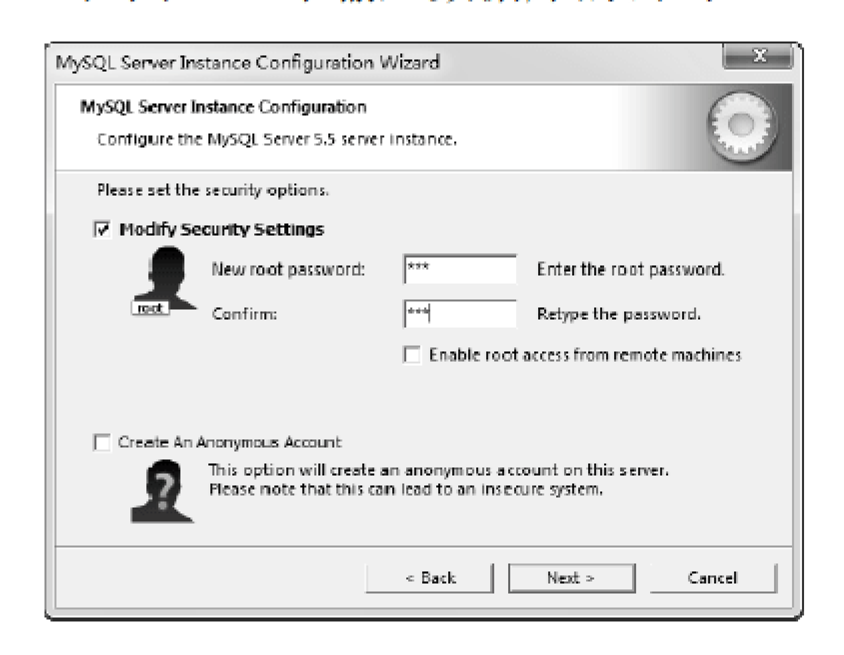


图 9.28　输入数据库的密码

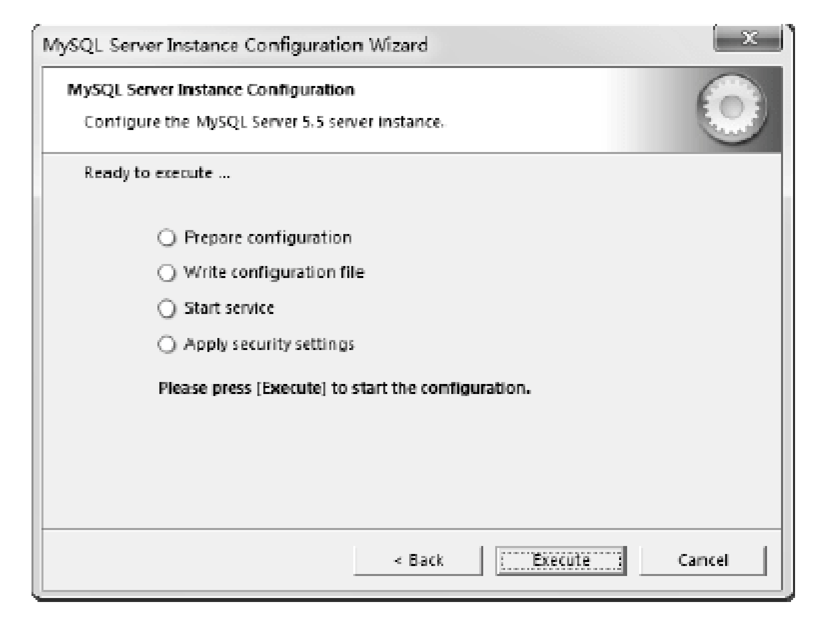


图 9.29　确认配置对话框

（20）在图 9.29 中，单击 Execute 按钮，执行前面进行的各项配置，如图 9.30 所示。在配置完成后，如图 9.31 所示。

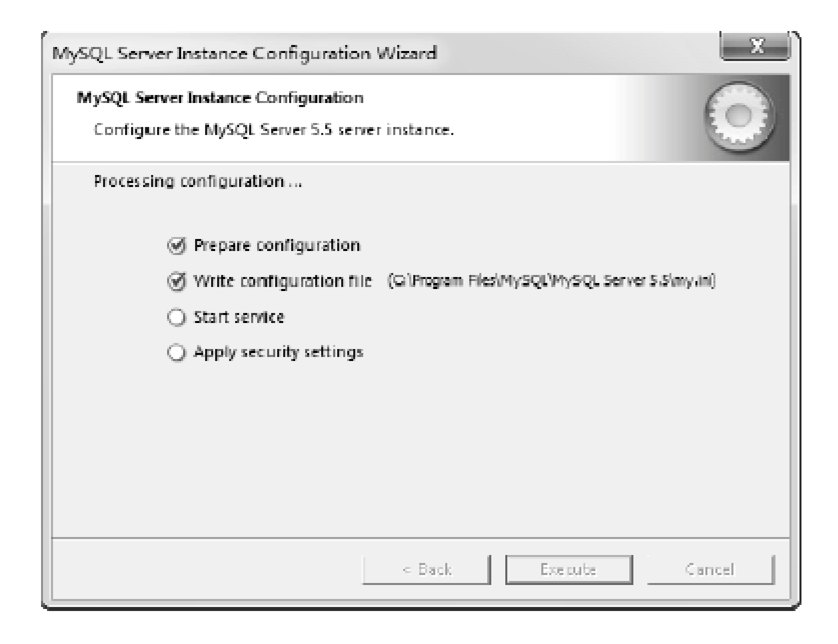


图 9.30　显示配置进度

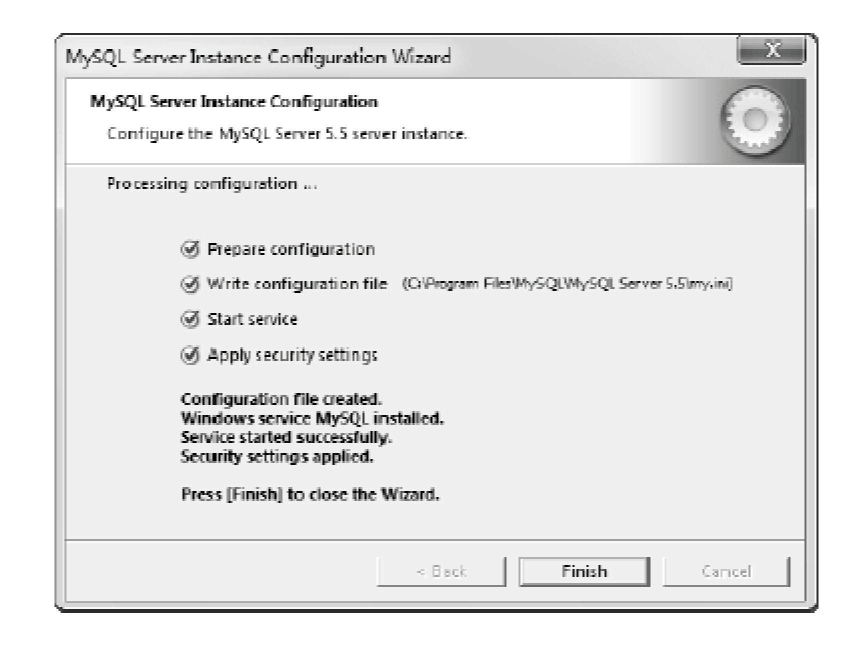


图 9.31　完成配置

到此 MySQL 安装成功，如果要查看 MySQL 的安装配置信息，则可以通过 MySQL 安装目录下的 my.ini 文件来完成。

在 my.ini 文件中，可以查看到 MySQL 服务器的端口号、MySQL 在本机的安装位置、MySQL 数据库文件存储的位置，以及 MySQL 数据库的编码等配置信息。参考内容如图 9.32 所示。

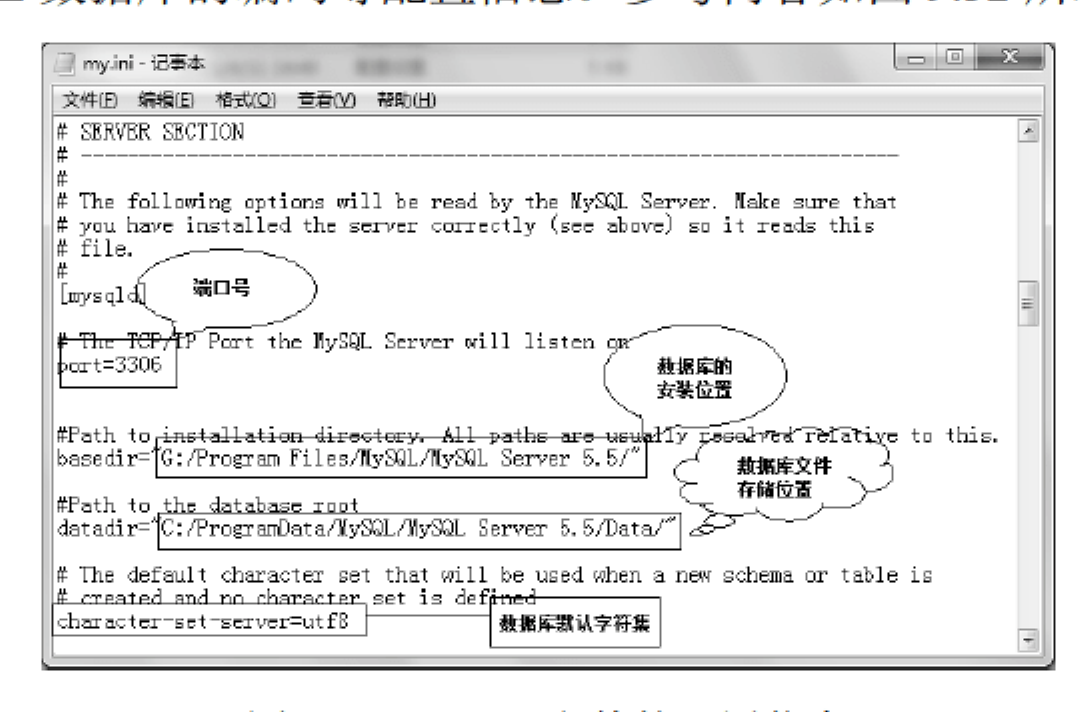


图 9.32　my.ini 文件的配置信息

9.4 启动、连接、断开和停止 MySQL 服务器

视频讲解：光盘\TM\Video\第 9 章\启动、连接、断开和停止 MySQL 服务器.exe

MySQL 数据库服务器的安装和维护相对简便，在 Linux/UNIX 操作系统中，使用终端命令即可实现对 MySQL 数据库服务器的管理，而在 Windows 操作系统中，不仅可以在命令提示符窗口中通过命令的方式管理 MySQL 服务器，还可以通过图形界面进行管理。下面将以 Windows 操作系统为例介绍如何对其进行启动、关闭等操作。

9.4.1 启动 MySQL 服务器

在 Windows 操作系统中启动 MySQL 服务器的方法主要有两种：通过系统服务方式和在命令提示符下通过命令行方式。

1. 通过系统服务启动 MySQL 服务器

如果 MySQL 设置为 Windows 服务，则可以通过选择“开始”/“管理工具”/“服务”命令，打开 Windows 服务管理器，在服务器的列表中找到 mysql 服务，单击鼠标右键，在弹出的快捷菜单中选择“启动”命令，启动 MySQL 服务，如图 9.33 所示。

2. 在命令提示符下启动 MySQL 服务器

选择“开始”/“运行”命令，在弹出的“运行”窗口中输入“cmd”命令，按 Enter 键进入命令提示符窗口。在命令提示符下输入：

```
\> net start mysql
```

按 Enter 键后，启用 MySQL 服务器，如图 9.34 所示。

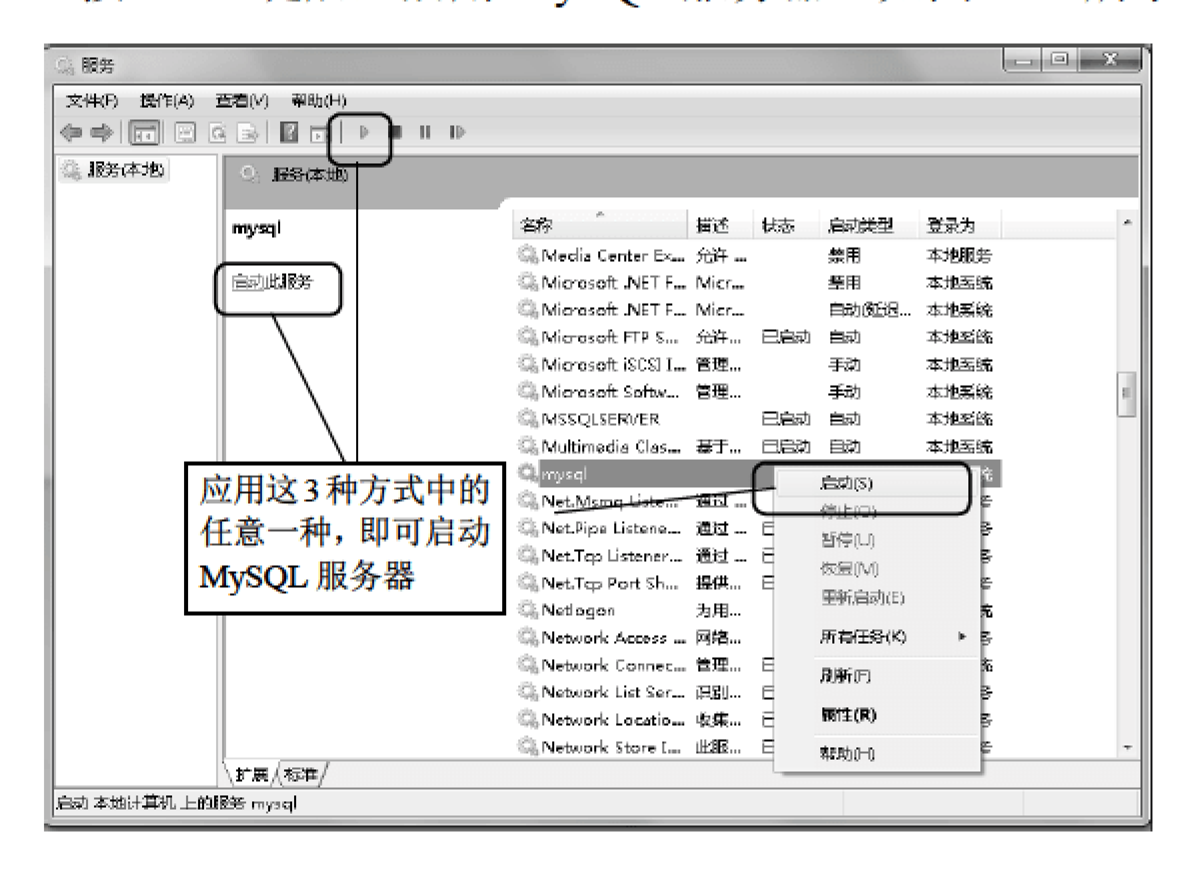


图 9.33　通过系统服务启动 MySQL 服务器

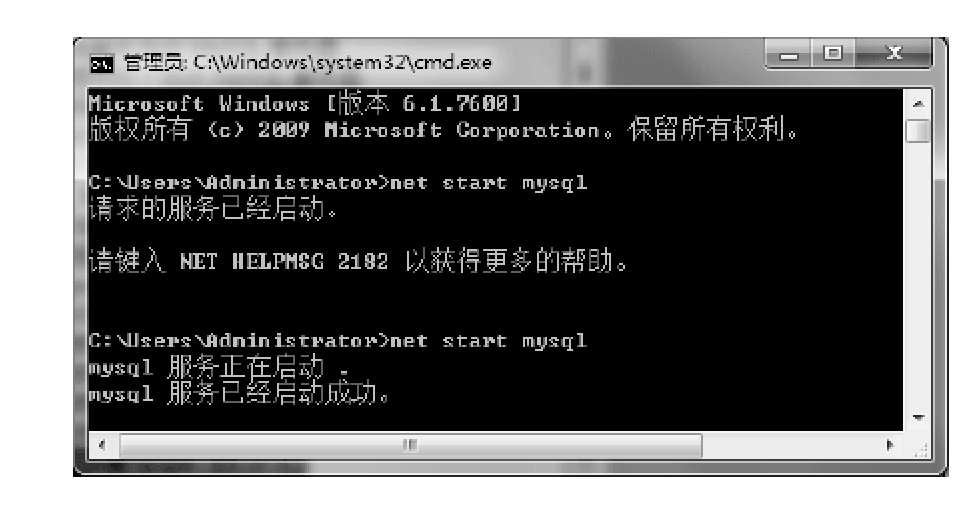
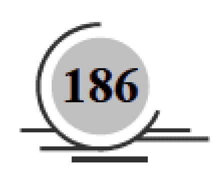


图 9.34　在命令提示符下启动 MySQL 服务器

9.4.2 连接和断开 MySQL 服务器

对 MySQL 数据库进行管理，首先要建立与 MySQL 服务器的连接。下面分别介绍连接和断开 MySQL 服务器的方法。



1．连接 MySQL 服务器

连接 MySQL 服务器通过 mysql 命令实现。在 MySQL 服务器启动后，选择“开始”/“运行”命令，在弹出的“运行”窗口中输入“cmd”命令，按 Enter 键后打开命令提示符窗口。在命令提示符下输入：

```
\> mysql  –h 主机名  –u 用户名  –p 密码
```

输入完命令语句后，按 Enter 键即可连接 MySQL 服务器，如图 9.35 所示。

2．断开 MySQL 服务器

连接到 MySQL 服务器后，可以通过在 MySQL 提示符下输入“exit”或者“quit”命令断开 MySQL 连接，如图 9.36 所示。

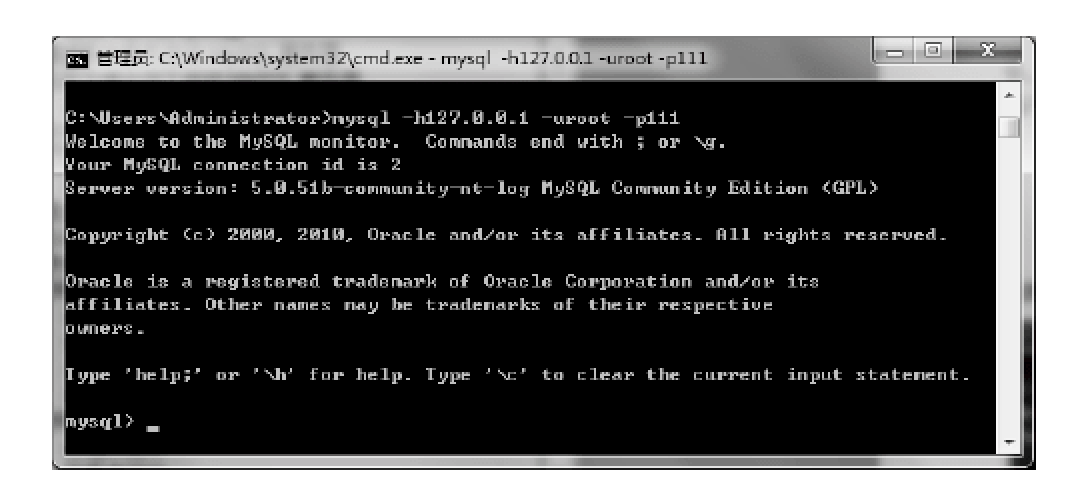


图 9.35　连接 MySQL 服务器

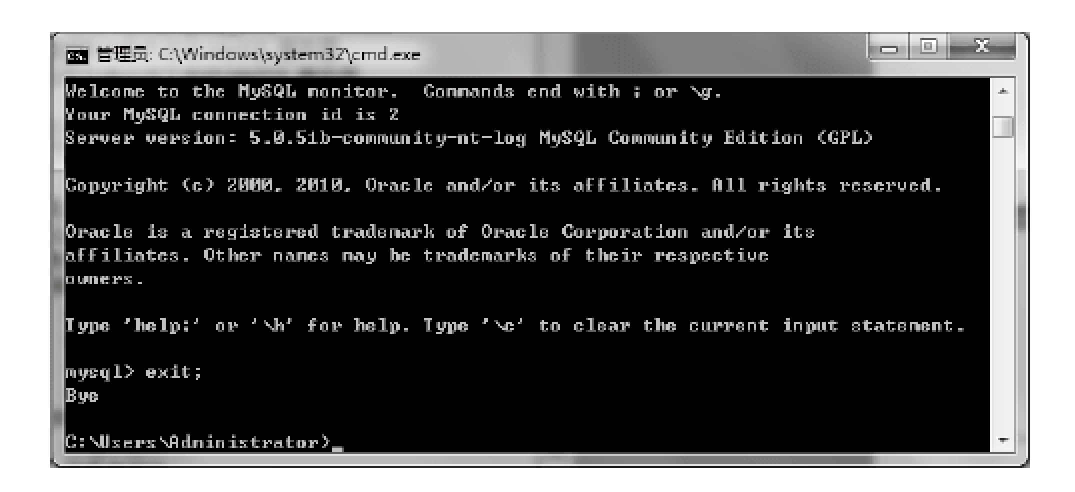


图 9.36　断开 MySQL 服务器

9.4.3　停止 MySQL 服务器

视频讲解：光盘\TM\Video\第 9 章\停止 MySQL 服务器.exe

为有效节省系统资源，在使用完 MySQL 服务后需要将其及时关闭。与启动 MySQL 服务类似，在 Windows 操作系统中，停止 MySQL 服务器主要有两种方法：通过系统服务关闭 MySQL 服务和命令提示符下通过命令行的方式关闭 MySQL 服务。下面具体介绍这两种实现方式。

1．通过系统服务停止 MySQL 服务器

如果使用的是安装版的 MySQL 数据库，则可以通过选择“开始”/“管理工具”/“服务”命令，打开 Windows 服务管理器，在服务器的列表中找到 mysql 服务并右击，在弹出的快捷菜单中选择“停止”命令，停止 MySQL 服务器，如图 9.37 所示。

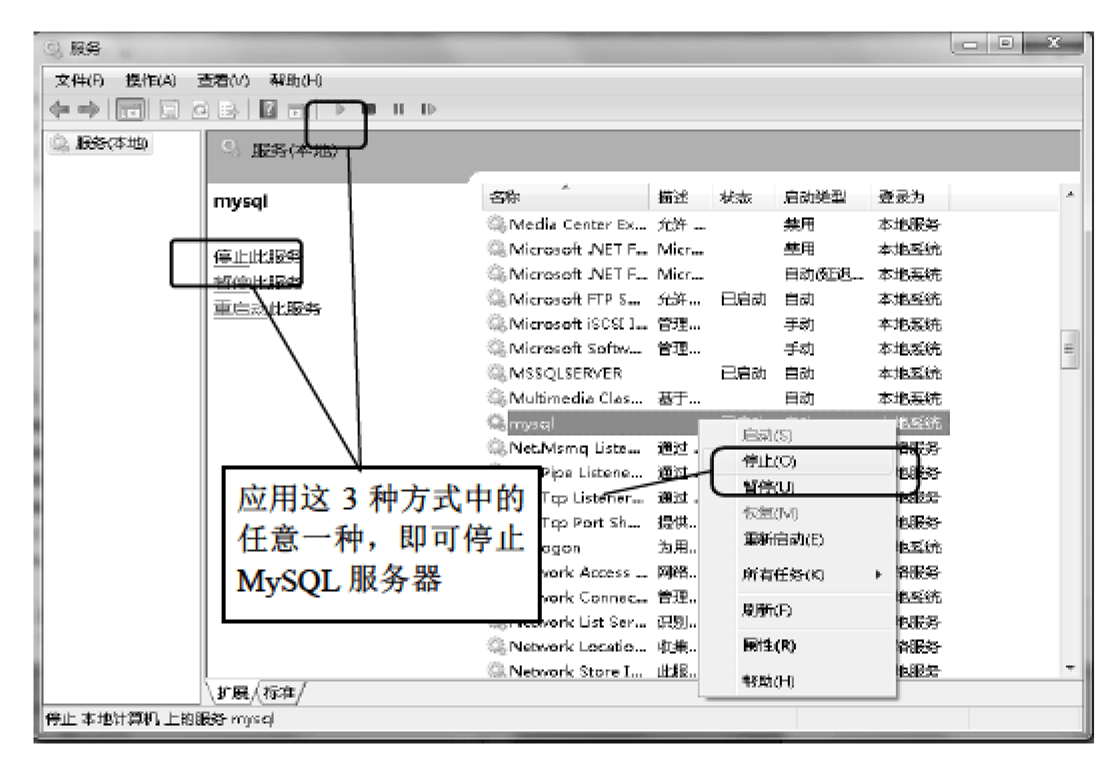


图 9.37　停止 MySQL 服务器

2．在命令提示符下停止 MySQL 服务器

选择“开始”/“运行”命令，输入“cmd”命令，进入命令提示符窗口，在命令提示符下输入：

```
\> net stop mysql
```

按 Enter 键即可停止 MySQL 服务器，如图 9.38 所示。

☑ 选择“开始”/“运行”命令，输入“cmd”命令，进入命令提示符窗口，在命令提示符下输入：

```
\ > mysqladmin –uroot shutdown –p111
```

按 Enter 键即可停止 MySQL 服务，如图 9.39 所示。



图 9.38　在命令提示符中停止 MySQL 服务器

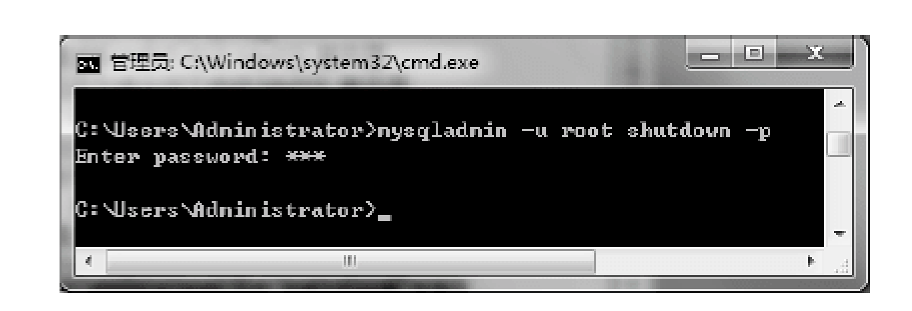


图 9.39　使用 mysqladmin 命令停止 MySQL 服务器

9.5　phpMyAdmin 图形化管理工具

phpMyAdmin 是众多 MySQL 图形化管理工具中应用最广泛的一种，是一款使用 PHP 开发的 B/S 模式的 MySQL 客户端软件，该工具是基于 Web 跨平台的管理程序，并且支持简体中文。用户可以在官方网站 www.phpMyAdmin.net 上免费下载到最新的版本。phpMyAdmin 为 Web 开发人员提供了类似于 Access、SQL server 的图形化数据库操作界面，通过该管理工具可以完全对 MySQL 进行操作，例如，创建数据库、数据表、生成 MySQL 数据库脚本文件等。

在浏览器地址栏中输入“http://localhost/phpMyAdmin/”，再在弹出的对话框中输入用户名和密码，进入 phpMyAdmin 图形化管理主界面，接下来就可以进行 MySQL 数据库的操作。下面将分别介绍如何创建、修改和删除数据库。

9.5.1　数据库操作管理

1．创建数据库

在 phpMyAdmin 的主界面，首先在文本框中输入数据库的名称 db_study，然后在下拉列表框中选择所要使用的编码，一般选择 gb2312_Chinese_ci（简体中文编码格式），单击“创建”按钮，创建数据库，如图 9.40 所示。成功创建数据库后，将显示如图 9.41 所示的界面。

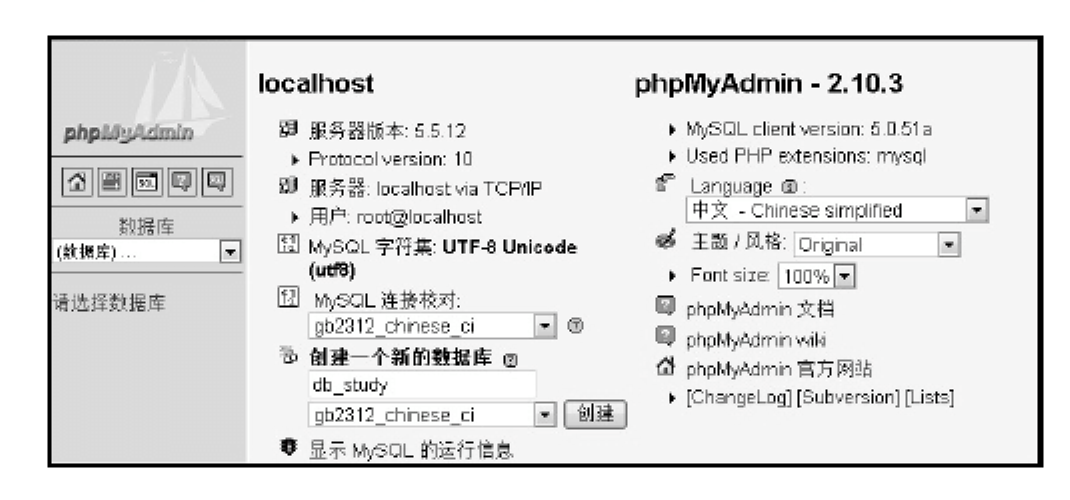


图 9.40　phpMyAdmin 管理主界面



图 9.41　成功创建数据库

2．修改数据库

在如图 9.41 所示的界面中，在右侧界面还可以对当前数据库进行修改。单击界面中的“操作”超链接，进入修改操作页面。

（1）可以对当前数据库执行创建数据表的操作，只要在创建数据表的提示信息下面的两个文本框中分



别输入要创建的数据表的名称和字段总数，然后单击“执行”按钮即可进入到创建数据表结构页面。

（2）也可以对当前的数据库重命名，在“重新命名数据库为”下的文本框中输入新的数据库名称，单击“执行”按钮，即可成功修改数据库名称。修改数据库名称如图 9.42 所示。

3．删除数据库

要删除某个数据库，首先在左侧的下拉菜单中选择该数据库，然后单击右侧界面中的删除超链接（如图 9.42 所示）即可成功删除指定的数据库。

9.5.2　管理数据表

视频讲解：光盘\TM\Video\第 9 章\管理数据表.exe

管理数据表是以选择指定的数据库为前题，然后在该数据库中创建并管理数据表。下面就来介绍如何创建、修改和删除数据表。

1．创建数据表

创建数据库 db_study 后，在右侧的操作页面中输入数据表的名称和字段数，然后单击“执行”按钮，即可创建数据表，如图 9.43 所示。

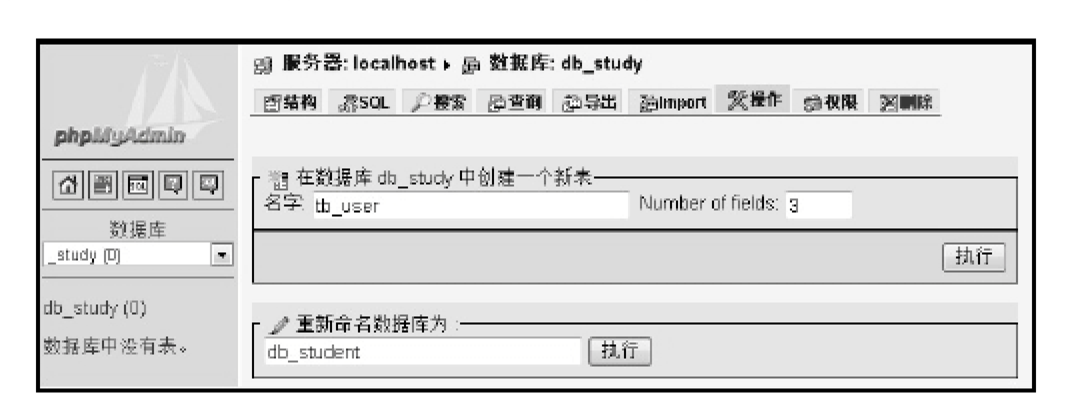


图 9.42　修改数据库

图 9.43　创建数据表

成功创建数据表 tb_admin 后，将显示数据表结构界面。在表单中对各个字段的详细信息进行输入，包括字段名、数据类型、长度/值、编码格式、是否为空、主键等，以完成对表结构的详细设置。当所有的信息都输入后，单击“保存”按钮，创建数据表结构，如图 9.44 所示。成功创建数据表结构后，将显示如图 9.45 所示的界面。



图 9.44　创建数据表结构



图 9.45　成功创建数据表



单击“保存”按钮，可以对数据表结构以横版显示进行表结构编辑。

2．修改数据表

一个新的数据表被创建后，进入到数据表页面中，在这里可以通过改变表的结构来修改表，可以执行添加新的列、删除列、索引列、修改列的数据类型或者字段的长度/值等操作，如图 9.46 所示。



图 9.46　修改数据表结构

3．删除数据表

要删除某个数据表，首先在左侧的下拉菜单中选择数据库，在指定的数据库中选择要删除的数据表，然后单击右侧界面中的删除超链接（如图 9.46 所示）即可成功删除指定的数据表。

9.5.3　管理数据记录

视频讲解：光盘\TM\Video\第 9 章\管理数据记录.exe

单击 phpMyAdmin 主界面中的SQL超链接，打开 SQL 语句编辑区。在编辑区输入完整的 SQL 语句，来实现数据的查询、添加、修改和删除操作。

1．使用 SQL 语句插入数据

在 SQL 语句编辑区应用 insert 语句向数据表 tb_admin 中插入数据后，单击“执行”按钮，便向数据表中插入了一条数据，如图 9.47 所示。如果提交的 SQL 语句有错误，系统会给出一个警告，提示用户修改它；如果提交的 SQL 语句正确，则弹出如图 9.48 所示的提示信息。

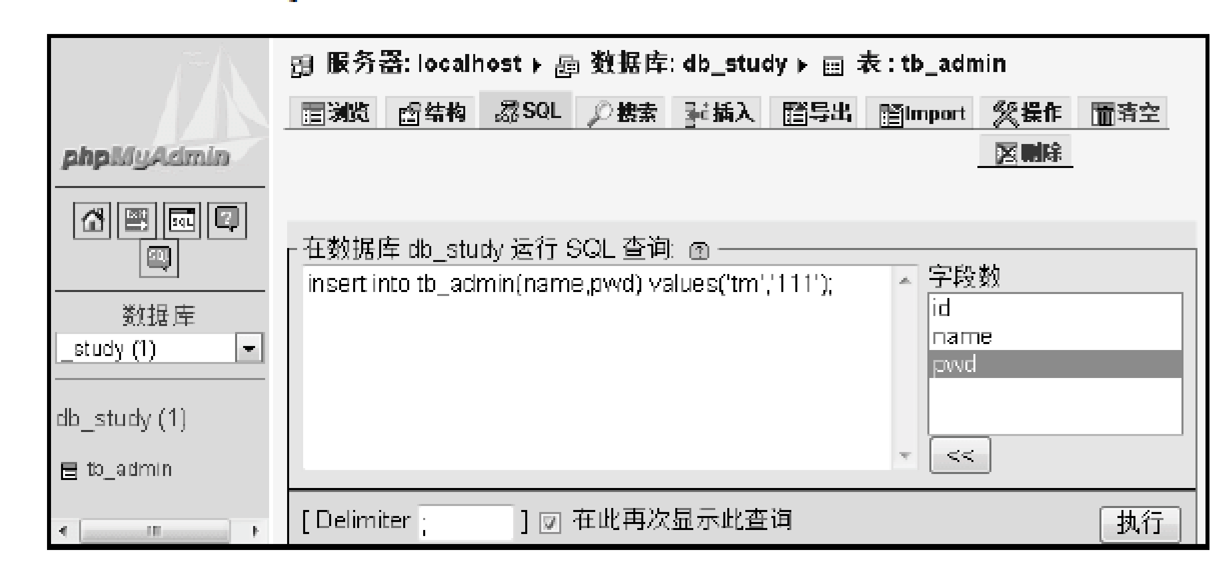


图 9.47　使用 SQL 语句向数据表中插入数据



图 9.48　成功添加数据信息



为了编写方便，可以利用其右侧的属性列表来选择要操作的列，只要选中要添加的列，双击其选项或者单击“<<”按钮添加列名称。

2. 使用 SQL 语句修改数据

在 SQL 语句编辑区应用 update 语句修改数据信息，将 ID 为 1 的管理员的名称改为纯净水，密码改为 111，添加的 SQL 语句如图 9.49 所示。

单击“执行”按钮，数据修改成功。比较修改前后的数据如图 9.50 所示。

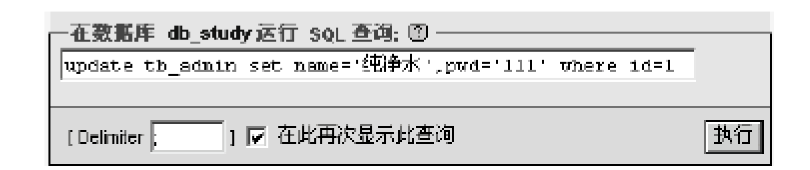


图 9.49　添加修改数据信息的 SQL 语句

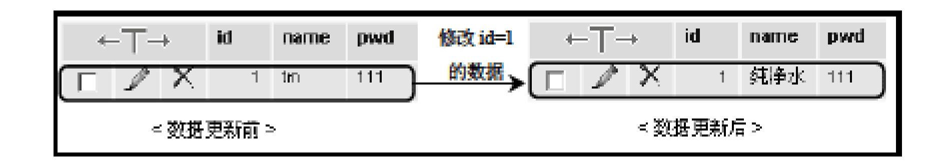


图 9.50　修改单条数据的实现过程

3. 使用 SQL 语句查询数据

在 SQL 语句编辑区应用 select 语句检索指定条件的数据信息，将 ID 小于 4 的管理员全部显示出来，添加的 SQL 语句如图 9.51 所示。

单击“执行”按钮，该语句的实现过程如图 9.52 所示。

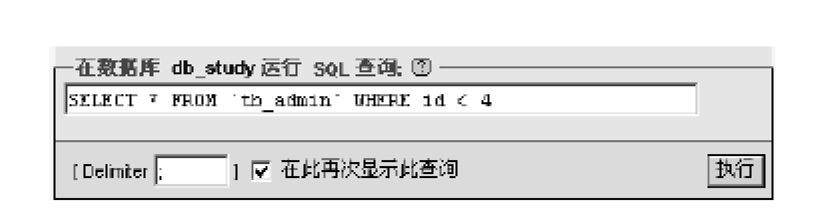


图 9.51　添加查询数据信息的 SQL 语句

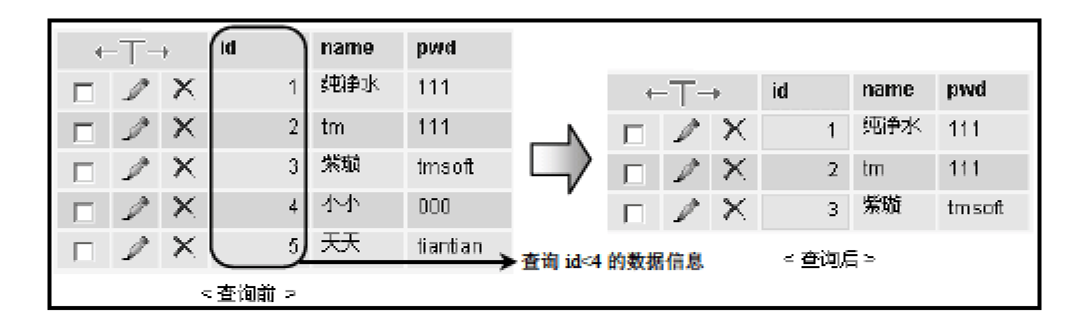


图 9.52　查询指定条件的实现过程

除了对整个表的简单查询外，还可以执行复杂的条件查询（使用 where 子句提交 LIKE、ORDER BY、GROUP BY 等条件查询语句）及多表查询，读者可通过上机进行实践，灵活运用 SQL 语句功能。

4. 使用 SQL 语句删除数据

在 SQL 语句编辑区应用 delete 语句删除指定条件的数据或全部数据信息，删除名称为 tm 的管理员信息，添加的 SQL 语句如图 9.53 所示。



如果 Delete 语句后面没有 Where 条件值，那么将删除指定数据表中的全部数据。

单击“执行”按钮，弹出确认删除操作对话框，单击“确定”按钮，执行数据表中指定条件的删除操作。该语句的实现过程如图 9.54 所示。

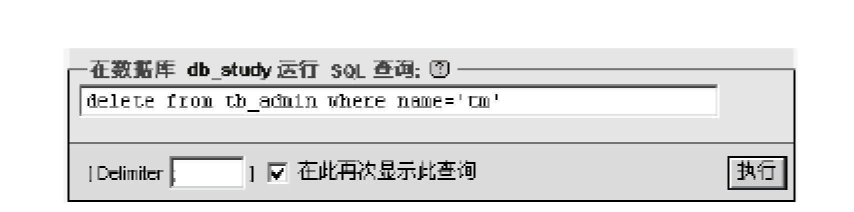


图 9.53　添加删除指定数据信息的 SQL 语句

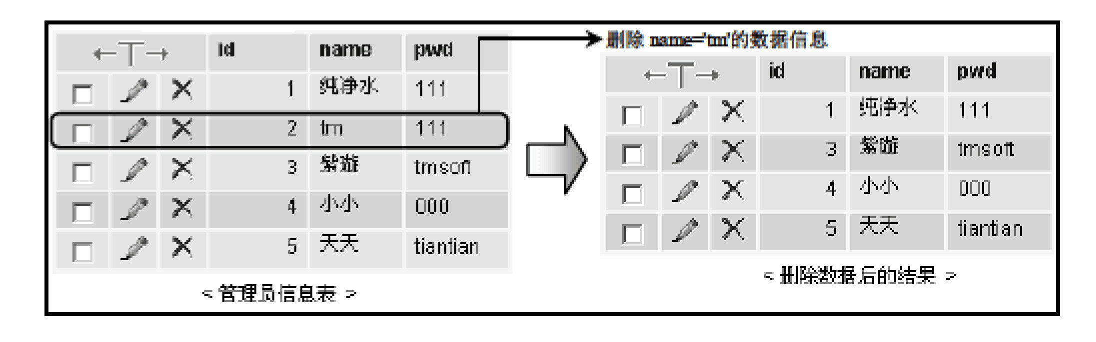


图 9.54　删除指定条件的实现过程

5. 通过 form 表单插入数据

选择某个数据表后，单击插入超链接，进入插入数据界面，如图 9.55 所示。在界面中输入各字段值，单击“执行”按钮即可插入记录。默认情况下，一次可以插入两条记录。

6. 浏览数据

选择某个数据表后，单击浏览超链接，进入浏览界面，如图 9.56 所示。单击每行记录中的按钮，

可以对该记录进行编辑；单击每行记录中的✕按钮，可以删除该条记录。

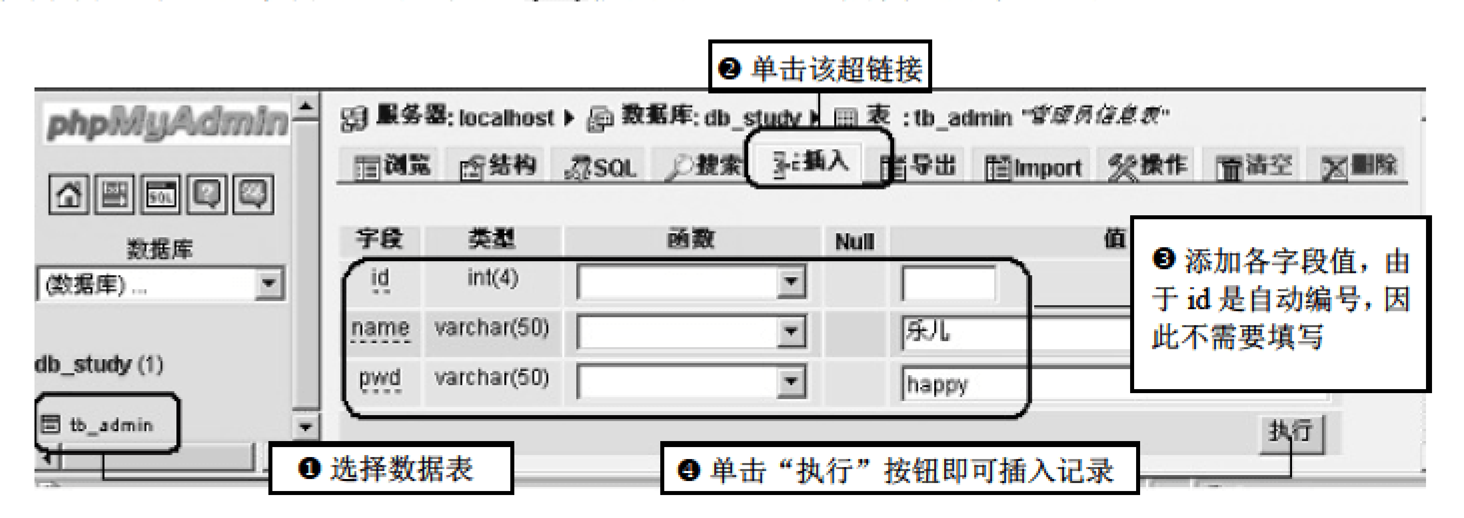


图 9.55　插入数据

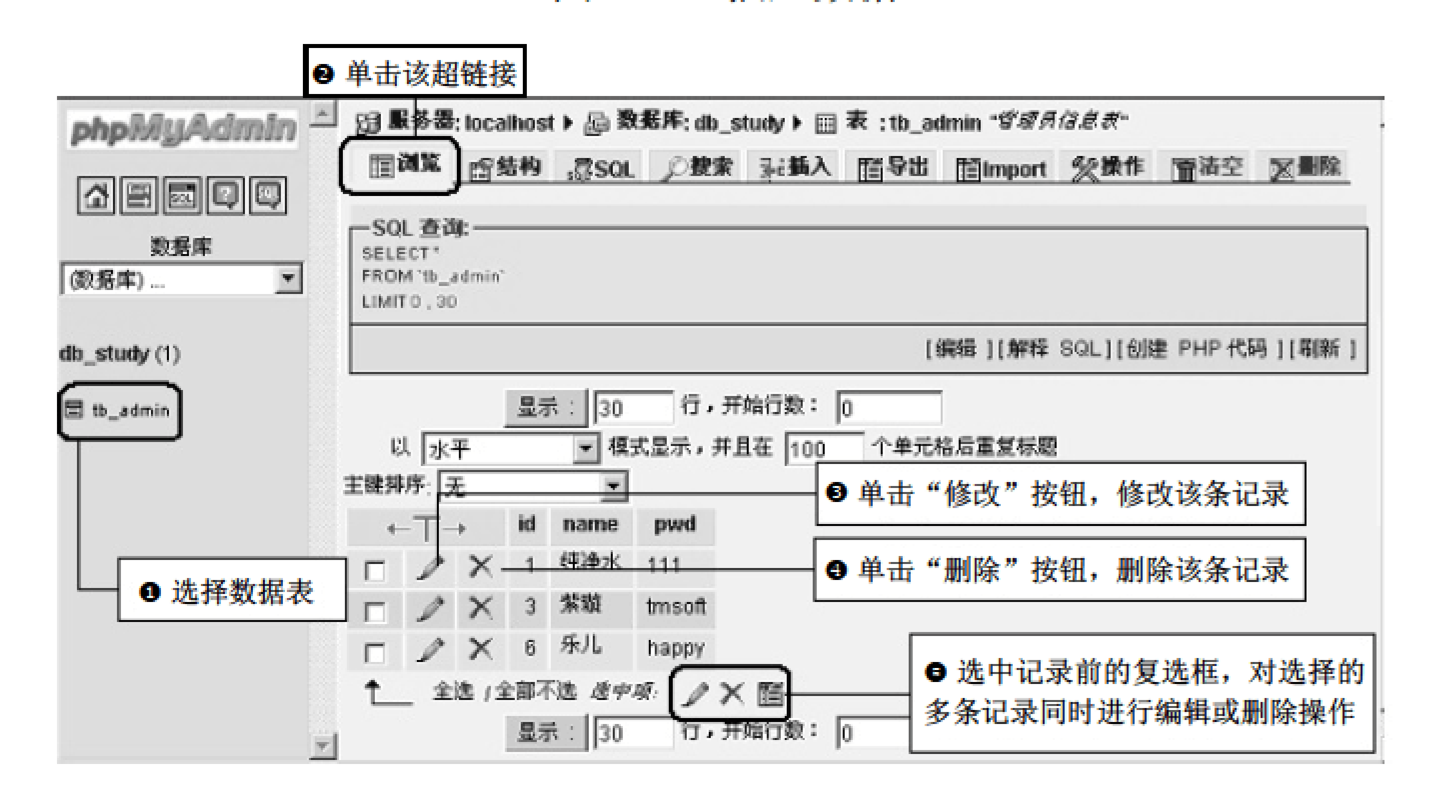


图 9.56　浏览数据

7．搜索数据

选择某个数据表后，单击搜索超链接，进入搜索页面，如图 9.57 所示。在这个页面中，可以在选择字段的列表框中选择一个或多个列，如果要选择多个列，先按下 Ctrl 键再单击要选择的字段名，查询结果将按照选择的字段名进行输出。

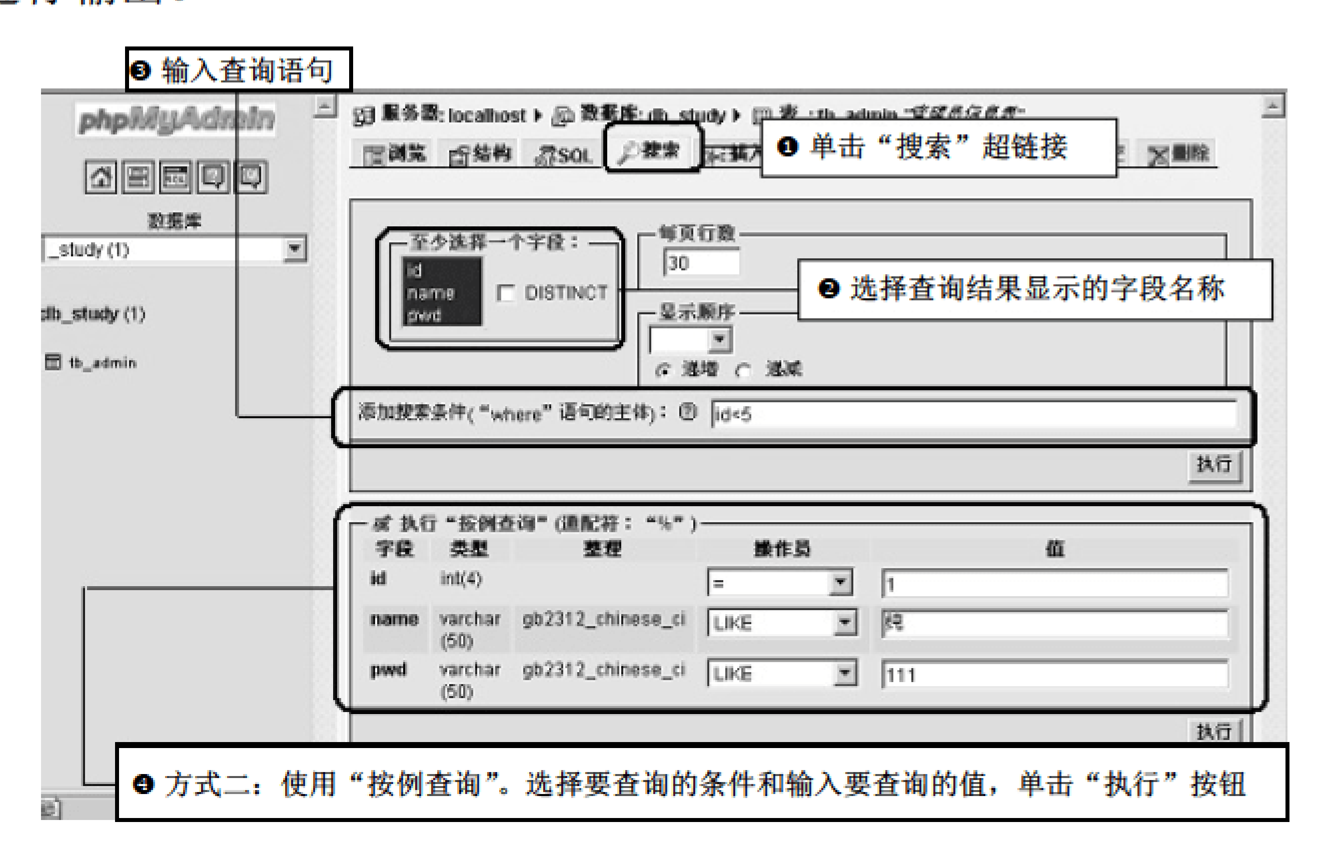


图 9.57　搜索查询

在该界面中可以对记录按条件进行查询。查询方式有两种：第一种方式选择构建 where 语句查询。直接在"where 语句的主体"文本框中输入查询语句，然后单击其后的"执行"按钮；第二种方式使用按例查询。选择查询的条件，并在文本框中输入要查询的值，单击"执行"按钮。



9.5.4　导入导出数据

视频讲解：光盘\TM\Video\第 9 章\导入导出数据.exe

导入和导出 MySQL 数据库脚本是互逆的两个操作。导入是执行扩展名为.sql 文件，将数据导入到数据库中；导出是将数据表结构、表记录存储为.sql 的脚本文件。通过导入和导出的操作实现数据库的备份和还原。

1．导出 MySQL 数据库脚本

单击 phpMyAdmin 主界面中的 导出 超链接，打开导出编辑区，如图 9.58 所示。选择导出文件的格式，这里默认使用选项 SQL，选中“另存为文件”复选框，单击“执行”按钮，将弹出如图 9.59 所示的文件下载对话框，单击“保存”按钮，将脚本文件以.sql 格式存储在指定位置。

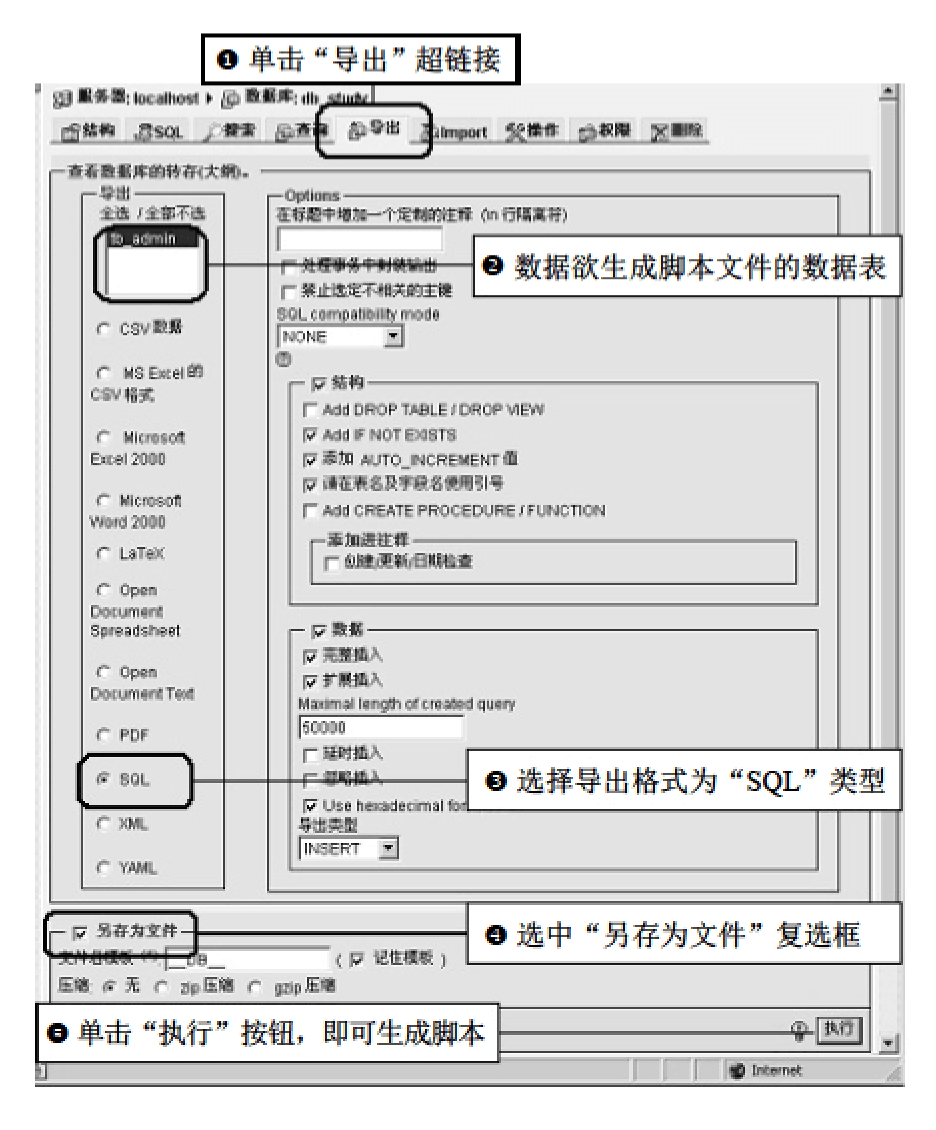


图 9.58　生成 MySQL 脚本文件设置界面

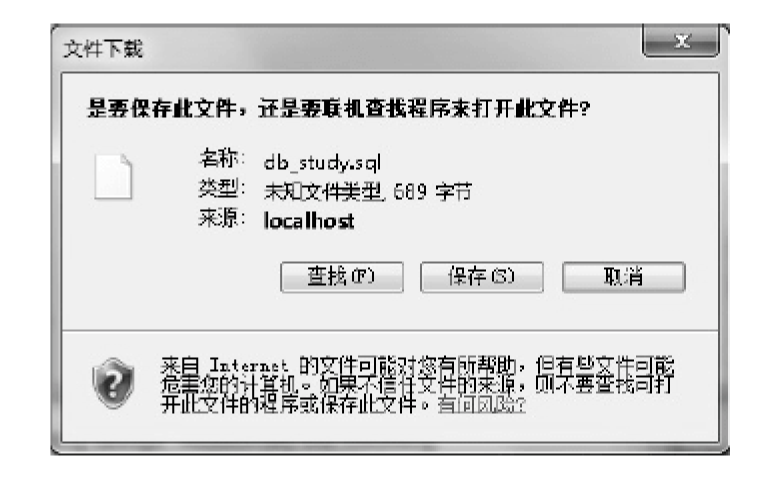


图 9.59　存储 MySQL 脚本对话框

2．导入 MySQL 数据库脚本

单击 Import 超链接，进入执行 MySQL 数据库脚本界面，单击“浏览”按钮查找脚本文件（如 db_study.sql）所在位置，如图 9.60 所示，单击“执行”按钮，即可执行 MySQL 数据库脚本文件。

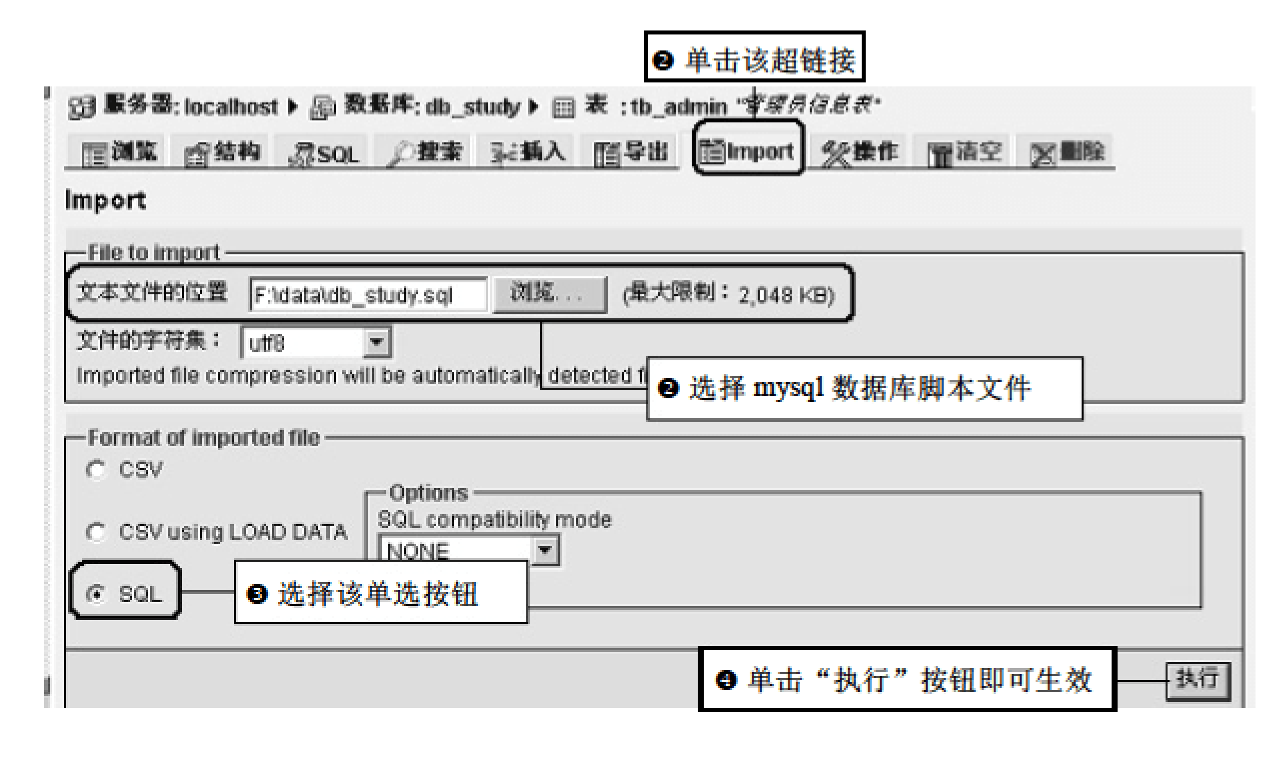


图 9.60　执行 MySQL 数据库脚本文件

注意 在执行 MySQL 脚本文件前，首先检测是否有与所导入同名的数据库，如果没有同名的数据库，则首先要在数据库中创建一个名称与数据文件中的数据库名相同的数据库，然后再执行 MySQL 数据库脚本文件。另外，在当前数据库中，不能有与将要导入数据库中的数据表重名的数据表存在，如果有重名的表存在，导入文件就会失败，提示错误信息。

说明 读者也可通过单击 phpMyAdmin 图形化工具左侧区的SQL按钮，在打开的对话框中，单击“导入文件”超链接，然后选择脚本文件所在的位置，从而执行脚本文件。

9.5.5 phpMyAdmin 设置编码格式

视频讲解：光盘\TM\Video\第 9 章\phpMyAdmin 设置编码格式.exe

将页面、程序文件、数据库与数据表设置统一的编码格式可以使程序运行时不至于出现乱码。一般情况下，设置页面的编码格式由 HTML 中的 meta 标签实现，设置程序文件的编码格式是由 header()函数实现，设置数据库与数据表的编码格式可以通过使用 phpMyAdmin 实现。下面以实例详细讲解一下如何为新创建的数据库设置编码格式。具体步骤如下：

（1）登录到 phpMyAdmin 图形化工具页面，创建数据库名称，并为新创建的数据库选择编码格式，如图 9.61 所示。

（2）创建数据表，定义数据表字段，并为新创建的数据表设置编码格式，如图 9.62 所示。



图 9.61 设置数据库编码格式



图 9.62 设置字段编码格式

9.5.6 phpMyAdmin 添加服务器新用户

视频讲解：光盘\TM\Video\第 9 章\phpMyAdmin 添加服务器新用户.exe

在 phpMyAdmin 图形化管理工具中，不但可以对 MySQL 数据库进行各种操作，而且可以添加服务器新用户，并对新添加的用户设置权限。



在 phpMyAdmin 中添加 MySQL 服务器新用户的步骤如下：

（1）单击 phpMyAdmin 主界面中的权限超链接，打开服务器用户操作界面，如图 9.63 所示。

（2）在该界面中，单击“添加新用户”按钮，进入到如图 9.64 所示界面，设置用户名、密码、主机和新用户的权限。设置完成后，单击“执行”按钮，完成对新用户的添加操作，返回主页面，将提示新用户添加成功。



图 9.63　服务器用户一览表

图 9.64　设置添加用户信息

9.5.7　phpMyAdmin 中重置 MySQL 服务器登录密码

视频讲解：光盘\TM\Video\第 9 章\phpMyAdmin 中重置 MySQL 服务器登录密码.exe

在 phpMyAdmin 图形化管理工具中，不但可以对 MySQL 数据库进行各种操作，而且可以对用户的权限进行设置，同时还可以对 MySQL 服务器的登录密码进行重置。

在 phpMyAdmin 中重置 MySQL 服务器登录密码的步骤如下：

（1）单击 phpMyAdmin 主界面中的权限超链接，打开服务器用户操作界面，如图 9.65 所示。

（2）在该界面中，可以对指定用户的权限进行编辑、可以添加新用户和删除指定的用户。这里选择指定的用户，单击（编辑权限）超链接，对指定用户的权限进行设置，进入到如图 9.66 所示的界面。



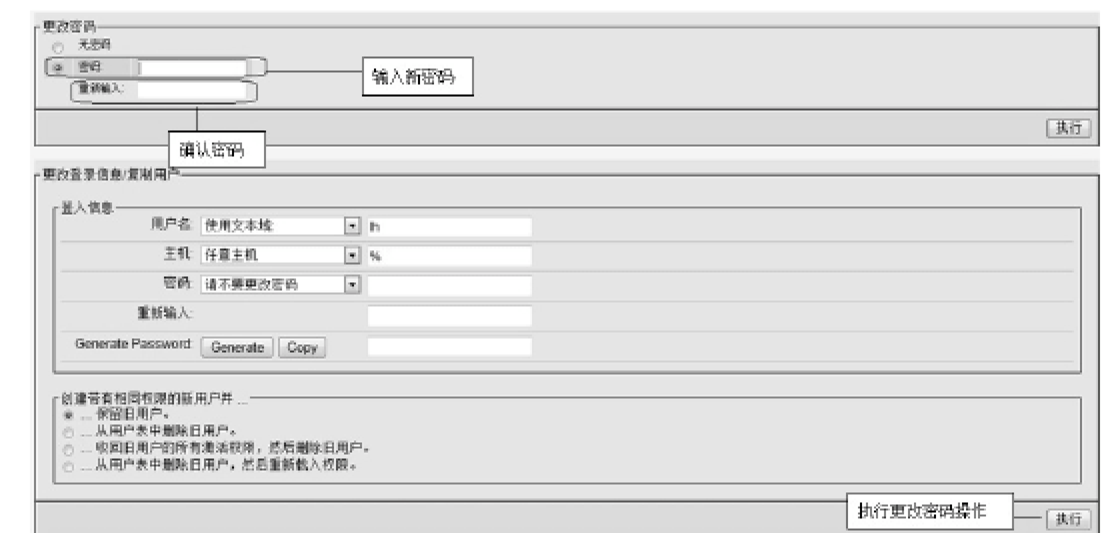


图 9.65　服务器用户一览表

图 9.66　编辑用户权限

在图 9.66 所示的界面中，可以设置用户的权限、修改密码、更改登录用户信息和复制用户。在输入新密码和确认密码后，单击“执行”按钮，完成对用户密码的修改操作，返回主页面，将提示密码修改成功。

9.6　小　　结

本章对 MySQL 数据库的基本概念、MySQL 5 的新特性进行了介绍，并详细介绍了 Windows 系统下使用“命令提示符”窗口创建和维护 MySQL 数据库和数据表的方法，在讲解过程中注重实践和常用命令的讲解，一切以能够帮助读者打好基础为出发点。通过本章的学习，读者能够了解 MySQL 数据库的基本操作和维护方法，掌握 MySQL 数据库中最基本和最常用命令的语法格式，并能够具备基本管理和维护 MySQL 数据库的能力。

9.7　学习成果检验

1．创建一个数据库 student，并在 student 中创建数据表 stu。完成表的创建后，向数据表中插入两条记录，然后删除第一条记录，最后显示表结构。**（答案位置：光盘\TM\Instance\09\9.1）**

2．在数据库 student 中，创建商品信息表 stu。**（答案位置：光盘\TM\Instance\09\9.2）**

3．创建数据库 stu 然后将 stu 更名为 student，并确保更名成功。**（答案位置：光盘\TM\Instance\09\9.3）**

第10章

MySQL 存储引擎与运算符

（视频讲解：33 分钟）

使用存储引擎可以加快查询的速度，并且每一种引擎都存在不同的含义。MySQL 的数据类型是数据的一种属性，它可以决定数据的存储格式、有效范围和相应的限制。并且可以让读者了解如何选择合适的数据类型，以及 MySQL 中讲述的运算符，其作用是用来指明对操作数所进行的运算。本章将介绍 MySQL 的存储引擎、数据类型的使用，以及各种运算符的详细讲解。

通过阅读本章内容，你可以：

- 了解 MySQL 存储引擎
- 了解 MySQL 数据类型、运算符
- 掌握算运算符的使用方法
- 掌握字符串和日期的数据类型
- 熟悉比较运算符的使用
- 熟悉逻辑运算符的使用

10.1 MySQL 存储引擎

存储引擎其实就是如何存储数据、如何为存储的数据建立索引和如何更新、查询数据等技术的实现方法。因为在关系数据库中数据的存储是以表的形式存储的，所以存储引擎也可以称为表类型（即存储和操作此表的类型）。在 Oracle 和 SQL Server 等数据库中只有一种存储引擎，所有数据存储管理机制都是一样的。而 MySQL 数据库提供了多种存储引擎。用户可以根据不同的需求为数据表选择不同的存储引擎，也可以根据自己的需要编写自己的存储引擎。

10.1.1 什么是 MySQL 存储引擎

MySQL 中的数据用各种不同的技术存储在文件（或者内存）中。这些技术中的每一种技术都使用不同的存储机制、索引技巧、锁定水平并且最终提供广泛的不同的功能和能力。通过选择不同的技术，你能够获得额外的速度或者功能，从而改善你的应用的整体功能。

这些不同的技术以及配套的相关功能在 MySQL 中被称作存储引擎（也称作表类型）。MySQL 默认配置了许多不同的存储引擎，可以预先设置或者在 MySQL 服务器中启用。你可以选择适用于服务器、数据库和表格的存储引擎，以便在选择如何存储信息、如何检索这些信息以及需要的数据结合什么性能和功能时为其提供最大的灵活性。

10.1.2 查询 MySQL 中支持的存储引擎

视频讲解：光盘\TM\Video\第 10 章\查询 MySQL 中支持的存储引擎.exe

使用 SHOW ENGINES 语句和 SHOW 语句可以查询 MySQL 中支持的存储引擎，下面分别使用这两个语句来查询 MySQL 中支持的存储引擎。

（1）使用 SHOW ENGINES 语句查询 MySQL 中支持的存储引擎。其查询语句如下：

```
SHOW ENGINES;
```

SHOW ENGINES 语句可以用“;”结束，也可以用“\g”或者“\G”结束。“\g”与“;”的作用是相同的，“\G”可以让结果显示得更加美观。使用 SHOW ENGINES 语句查询的查询结果如图 10.1 所示。

查询结果中的 Engine 参数指的是存储引擎的名称；Support 参数指的是 MySQL 是否支持该类引擎，YES 表示支持；Comment 参数指对该引擎的评论。

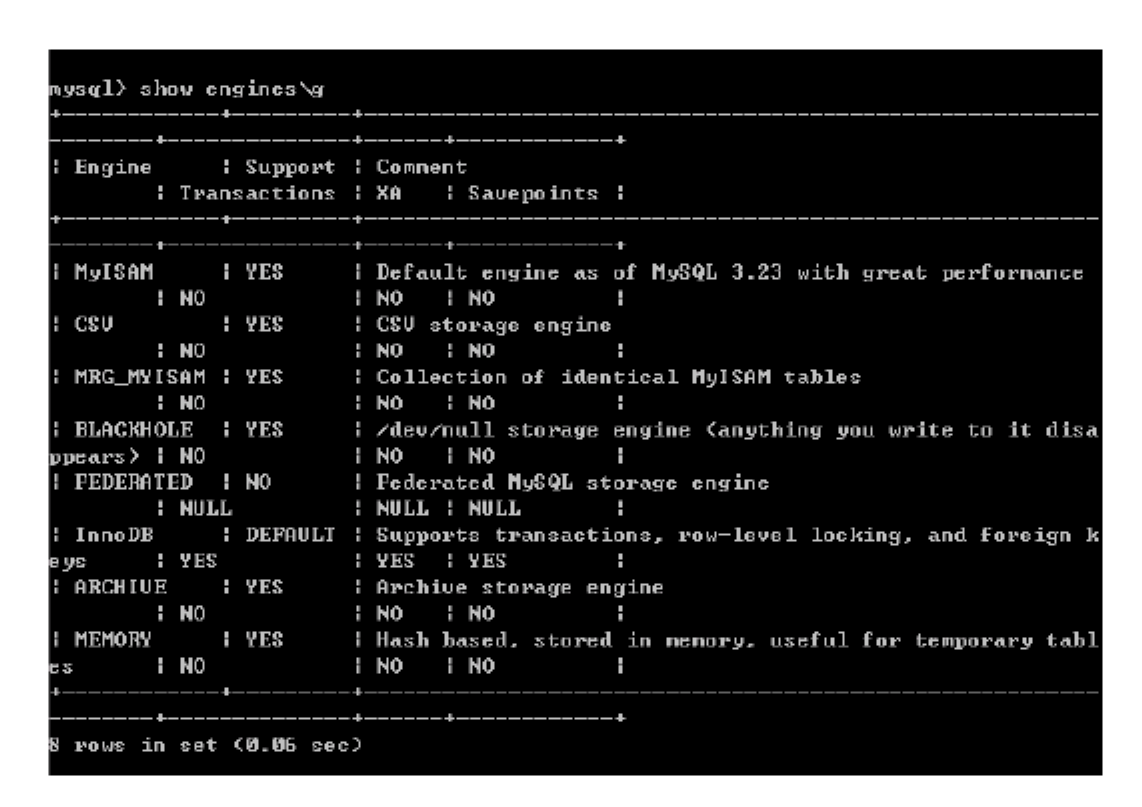
从查询结果中可以看出，MySQL 支持多个存储引擎，其中 InnoDB 为默认存储引擎。

（2）使用 SHOW 语句查询 MySQL 中支持的存储引擎。其查询语句如下：

```
SHOW VARIABLES LIKE 'have%';
```

使用 SHOW VARIABLES 语句查询的结果如图 10.2 所示。

有些表根本不用来存储长期数据，实际上用户需要完全在服务器的 RAM 或特殊的临时文件中创建和维护这些数据，以确保高性能，但这样以来，也存在很高的不稳定风险。还有一些表只是为了简化对一组相同表的维护和访问，为同时与所有这些表交互提供一个单一接口。另外还有其他一些特别用途的表，但重点是：MySQL 支持很多类型的表，每种类型都有自己特定的作用、优点和缺点。MySQL 还相应地提供了很多不同的存储引擎，可以以最适合于应用需求的方式存储数据。MySQL 有多个可用的存储引擎，下面主要介绍 InnoDB、MyISAM 和 MEMEORY 3 种存储引擎。



```
mysql> show engines\g
+------------+---------+------------------------------------------------------------
---------+--------------+------+------------+
| Engine     | Support | Comment
         | Transactions | XA   | Savepoints |
+------------+---------+------------------------------------------------------------
---------+--------------+------+------------+
| MyISAM     | YES     | Default engine as of MySQL 3.23 with great performance
         | NO           | NO   | NO         |
| CSV        | YES     | CSV storage engine
         | NO           | NO   | NO         |
| MRG_MYISAM | YES     | Collection of identical MyISAM tables
         | NO           | NO   | NO         |
| BLACKHOLE  | YES     | /dev/null storage engine (anything you write to it disa
ppears) | NO           | NO   | NO         |
| FEDERATED  | NO      | Federated MySQL storage engine
         | NULL         | NULL | NULL       |
| InnoDB     | DEFAULT | Supports transactions, row-level locking, and foreign k
eys      | YES          | YES  | YES        |
| ARCHIVE    | YES     | Archive storage engine
         | NO           | NO   | NO         |
| MEMORY     | YES     | Hash based, stored in memory, useful for temporary tabl
es       | NO           | NO   | NO         |
+------------+---------+------------------------------------------------------------
---------+--------------+------+------------+
8 rows in set (0.06 sec)
```

图 10.1　使用 SHOW ENGINES 语句查询 MySQL 中支持的存储引擎

```
mysql> show variables like 'have%';
+-------------------------+----------+
| Variable_name           | Value    |
+-------------------------+----------+
| have_community_features | YES      |
| have_compress           | YES      |
| have_crypt              | NO       |
| have_csv                | YES      |
| have_dynamic_loading    | YES      |
| have_geometry           | YES      |
| have_innodb             | YES      |
| have_ndbcluster         | NO       |
| have_openssl            | DISABLED |
| have_partitioning       | YES      |
| have_query_cache        | YES      |
| have_rtree_keys         | YES      |
| have_ssl                | DISABLED |
| have_symlink            | YES      |
+-------------------------+----------+
14 rows in set (0.01 sec)
```

图 10.2　使用 SHOW 语句查询 MySQL 中支持的存储引擎

10.1.3　InnoDB 存储引擎

InnoDB 已经开发了十余年，遵循 CNU 通用公开许可（GPL）发行。InnoDB 已经被一些重量级因特网公司所采用，如 Yahoo!、Slashdot 和 Google，为用户操作非常大的数据库提供了一个强大的解决方案。InnoDB 给 MySQL 的表提供了事务、回滚、崩溃修复能力和多版本并发控制的事务安全。MySQL 从 3.23.34a 开始包含 InnoDB 存储引擎。InnoDB 是 MySQL 上第一个提供外键约束的表引擎，而且 InnoDB 对事务处理的能力，也是 MySQL 其他存储引擎所无法与之比拟的。下面介绍 InnoDB 存储引擎的特点及其优缺点。

InnoDB 存储引擎中支持自动增长列 AUTO_INCREMENT。自动增长列的值不能为空，且值必须唯一。MySQL 中规定自增列必须为主键。在插入值时，如果自动增长列不输入值，则插入的值为自动增长后的值；如果输入的值为 0 或空（NULL），则插入的值也为自动增长后的值；如果插入某个确定的值，且该值在前面没有出现过，则可以直接插入。

InnoDB 存储引擎中支持外键（FOREIGN KEY）。外键所在的表为子表，外键所依赖的表为父表。父表中被子表外键关联的字段必须为主键。当删除、更新父表的某条信息时，子表也必须有相应的改变。InnoDB 存储引擎中，创建的表的表结构存储在.frm 文件中。数据和索引存储在 innodb_data_home_dir 和 innodb_data_file_path 表空间中。

InnoDB 存储引擎的优势在于提供了良好的事务管理、崩溃修复能力和并发控制。缺点是其读写效率稍差，占用的数据空间相对比较大。

InnoDB 表是如下情况的理想引擎。

☑　更新密集的表：InnoDB 存储引擎特别适合处理多重并发的更新请求。

☑　事务：InnoDB 存储引擎是唯一支持事务的标准 MySQL 存储引擎，这是管理敏感数据（如金融信息和用户注册信息）的必需软件。

☑　自动灾难恢复：与其他存储引擎不同，InnoDB 表能够自动从灾难中恢复。虽然 MyISAM 表是能在灾难后修复，但其过程要长得多。

Oracle 的 InnoDB 存储引擎广泛应用于基于 MySQL 的 Web、电子商务、金融系统、健康护理以及零售应用。因为 InnoDB 可提供高效的 ACID 独立性（Atomicity）、一致性（Consistency）、隔离性（Isolation）、持久性（Durability）兼容事务处理能力，以及独特的高性能和具有可扩展性的构架要素。

另外，InnoDB 设计用于事务处理应用，这些应用需要处理崩溃恢复、参照完整性、高级别的用户并发数，以及响应时间超时服务水平合同。在 MySQL 5.5 中，最显著的增强性能是将 InnoDB 作为默认的存储

引擎。在 MyISAM 以及其他表类型依然可用的情况下，用户无须更改配置，就可构建基于 InnoDB 的应用程序。

10.1.4 MyISAM 存储引擎

MyISAM 存储引擎是 MySQL 中常见的存储引擎，曾是 MySQL 的默认存储引擎。MyISAM 存储引擎是基于 ISAM 存储引擎发展起来的，它解决了 ISAM 的很多不足。MyISAM 增加了很多有用扩展。

1．MyISAM 存储引擎的文件类型

MyISAM 存储引擎的表存储成 3 个文件。文件的名字与表名相同。扩展名包括 frm、myd 和 myi。

☑ frm：存储表的结构。
☑ myd：存储数据，是 MYData 的缩写。
☑ myi：存储索引，是 MYIndex 的缩写。

2．MyISAM 存储引擎的存储格式

基于 MyISAM 存储引擎的表支持 3 种不同的存储格式，包括静态型、动态型和压缩型。

（1）MyISAM 静态

如果所有表列的大小都是静态的（即不使用 xBLOB、xTEXT 或 VARCHAR 数据类型），MySQL 就会自动使用静态 MyISAM 格式。使用这种类型的表性能非常高，因为在维护和访问以预定义格式存储的数据时需要很低的开销。但是，这项优点要以空间为代价，因为每列都需要分配该列的最大空间，而无论该空间是否真正地使用。

（2）MyISAM 动态

如果有表列（即使只有一列）定义为动态的（使用 xBLOB、xTEXT 或 VARCHAR），MySQL 就会自动使用动态格式。虽然 MyISAM 动态表占用的空间比静态格式所占空间少，但空间的节省带来了性能的下降。如果某个字段的内容发生改变，其位置很可能就需要移动，这会导致碎片的产生。随着数据集中的碎片增加，数据访问性能就会相应降低。这个问题有两种修复方法：

☑ 尽可能使用静态数据类型。
☑ 经常使用 OPTIMIZE TABLE 语句，它会整理表的碎片，恢复由于表更新和删除而导致的空间丢失。

（3）MyISAM 压缩

有时会创建在整个应用程序生命周期中都只读的表。如果是这种情况，就可以使用 myisampack 工具将其转换为 MyISAM 压缩表来减少空间。在给定硬件配置下（如快速的处理器和低速的硬盘驱动器），性能的提升将相当显著。

3．MyISAM 存储引擎的优缺点

MyISAM 存储引擎的优势在于占用空间小，处理速度快。缺点是不支持事务的完整性和并发性。

10.1.5 MEMORY 存储引擎

MEMORY 存储引擎是 MySQL 中的一类特殊的存储引擎。其使用存储在内存中的内容来创建表，而且所有数据也放在内存中。这些特性都与 InnoDB 存储引擎、MyISAM 存储引擎不同。下面将对 MEMORY 存储引擎的文件存储形式、索引类型、存储周期和优缺点等进行讲解。

1．MEMORY 存储引擎的文件存储形式

每个基于 MEMORY 存储引擎的表实际对应一个磁盘文件。该文件的文件名与表名相同，类型为 frm。



该文件中只存储表的结构。而其数据文件，都是存储在内存中。这样有利于对数据的快速处理，提高整个表的处理效率。值得注意的是，服务器需要有足够的内存来维持 MEMORY 存储引擎的表的使用。如果不需要使用了，可以释放这些内容，甚至可以删除不需要的表。

2．MEMORY 存储引擎的索引类型

MEMORY 存储引擎默认使用哈希（HASH）索引。其速度要比使用 B 型树（BTREE）索引快。如果读者希望使用 B 型树索引，可以在创建索引时选择使用。

3．MEMORY 存储引擎的存储周期

MEMORY 存储引擎通常很少用到。因为 MEMORY 表的所有数据是存储在内存上的，如果内存出现异常就会影响到数据的完整性。如果重启机器或者关机，表中的所有数据将消失。因此，基于 MEMORY 存储引擎的表生命周期很短，一般都是一次性的。

4．MEMORY 存储引擎的优缺点

MEMORY 表的大小是受到限制的。表的大小主要取决于两个参数，分别是 max_rows 和 max_heap_table_size。其中，max_rows 可以在创建表时指定；max_heap_table_size 的大小默认为 16MB，可以按需要进行扩大。因此，其存在于内存中的特性，这类表的处理速度非常快。但是，其数据易丢失，生命周期短。

创建 MySQL MEMORY 存储引擎的出发点是速度。为得到最快的响应时间，采用的逻辑存储介质是系统内存。虽然在内存中存储表数据确实会提高性能，但要记住，当 mysqld 守护进程崩溃时，所有的 MEMORY 数据都会丢失。

MEMORY 表不支持 VARCHAR、BLOB 和 TEXT 数据类型，因为这种表类型按固定长度的记录格式存储。此外，如果使用版本 4.1.0 之前的 MySQL，则不支持自动增加列（通过 AUTO_INCREMENT 属性）。当然，要记住 MEMORY 表只用于特殊的范围，不会用于长期存储数据。基于其这个缺陷，选择 MEMORY 存储引擎时要特别小心。

当数据有如下情况时，可以考虑使用 MEMORY 表。

- ☑ 暂时：目标数据只是临时需要，在其生命周期中必须立即可用。
- ☑ 相对无关：存储在 MEMORY 表中的数据如果突然丢失，不会对应用服务产生实质的负面影响，而且不会对数据完整性有长期影响。

如果使用 MySQL4.1 及以之前版本，MEMORY 的搜索比 MyISAM 表的搜索效果要低，因为 MEMORY 表只支持散列索引，这需要使用整个键进行搜索。但是，4.1 之后的版本同时支持散列索引和 B 树索引。B 树索引优于散列索引的是，可以使用部分查询和通配查询，也可以使用“<”、“>”和“>=”等操作符方便数据挖掘。

10.1.6　如何选择存储引擎

每种存储引擎都有各自的优势，不能笼统地说谁比谁更好，只有适合不适合。下面根据其不同的特性，给出选择存储引擎的建议。

- ☑ InnoDB 存储引擎：用于事务处理应用程序，具有众多特性，包括 ACID 事务支持，支持外键。同时支持崩溃修复能力和并发控制。如果需要对事务的完整性要求比较高，要求实现并发控制，那选择 InnoDB 存储引擎有其很大的优势。如果需要频繁地进行更新、删除操作的数据库，也可以选择 InnoDB 存储引擎。因为，该类存储引擎可以实现事务的提交（Commit）和回滚（Rollback）。
- ☑ MyISAM 存储引擎：管理非事务表，它提供高速存储和检索，以及全文搜索能力。MyISAM 存储引擎插入数据快，空间和内存使用比较低。如果表主要是用于插入新记录和读出记录，那么选择 MyISAM 存储引擎能实现处理的高效率。如果应用的完整性、并发性要求很低，也可以选择

MyISAM 存储引擎。

☑ MEMORY 存储引擎：MEMORY 存储引擎提供“内存中”表，MEMORY 存储引擎的所有数据都在内存中，数据的处理速度快，但安全性不高。如果需要很快的读写速度，对数据的安全性要求较低，可以选择 MEMORY 存储引擎。MEMORY 存储引擎对表的大小有要求，不能建太大的表。所以，这类数据库只使用相对较小的数据库表。

以上存储擎引的选择建议是根据不同存储引擎的特点提出的，并不是绝对的。实际应用中还需要根据各自的实际情况进行分析。

10.1.7 设置数据表的存储引擎

视频讲解：光盘\TM\Video\第 10 章\设置数据表的存储引擎.exe

下面创建 db_database03 数据库文件，在数据库中创建三个数据表，并分别为其设置不同的存储引擎。以此来诠释这三种不同存储引擎创建的数据表文件有什么区别。

（1）创建 tb_001 数据表，设置存储引擎为 MyISAM，生成的数据表文件如图 10.3 所示，由 3 个不同后缀的文件组成。



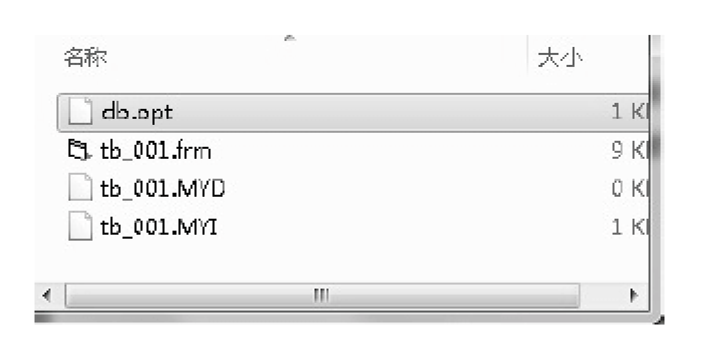
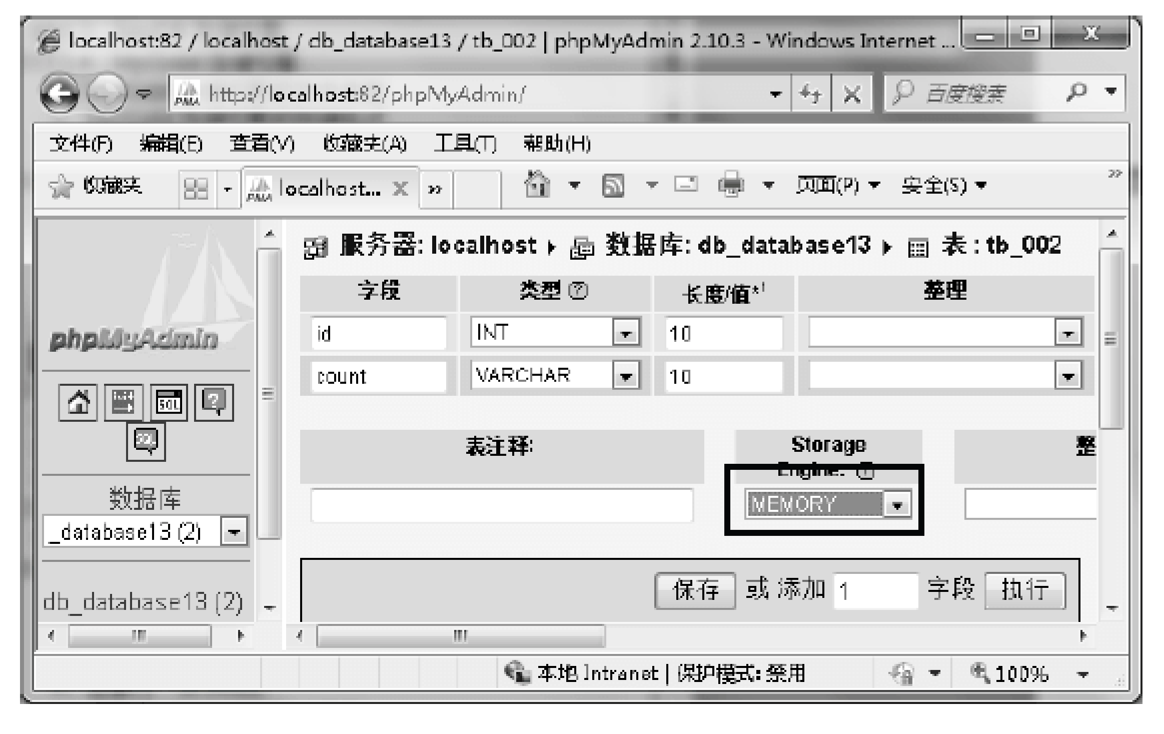


图 10.3 创建 tb_001 数据表及生成的数据表文件

（2）创建 tb_002 数据表，设置存储引擎为 MEMORY，生成的数据表文件如图 10.4 所示，只有一个.frm 为后缀的文件。



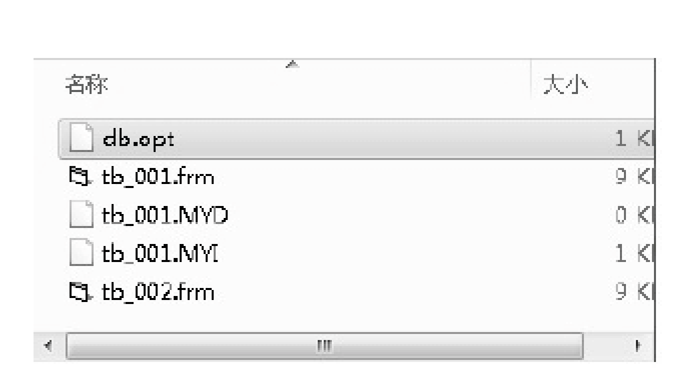
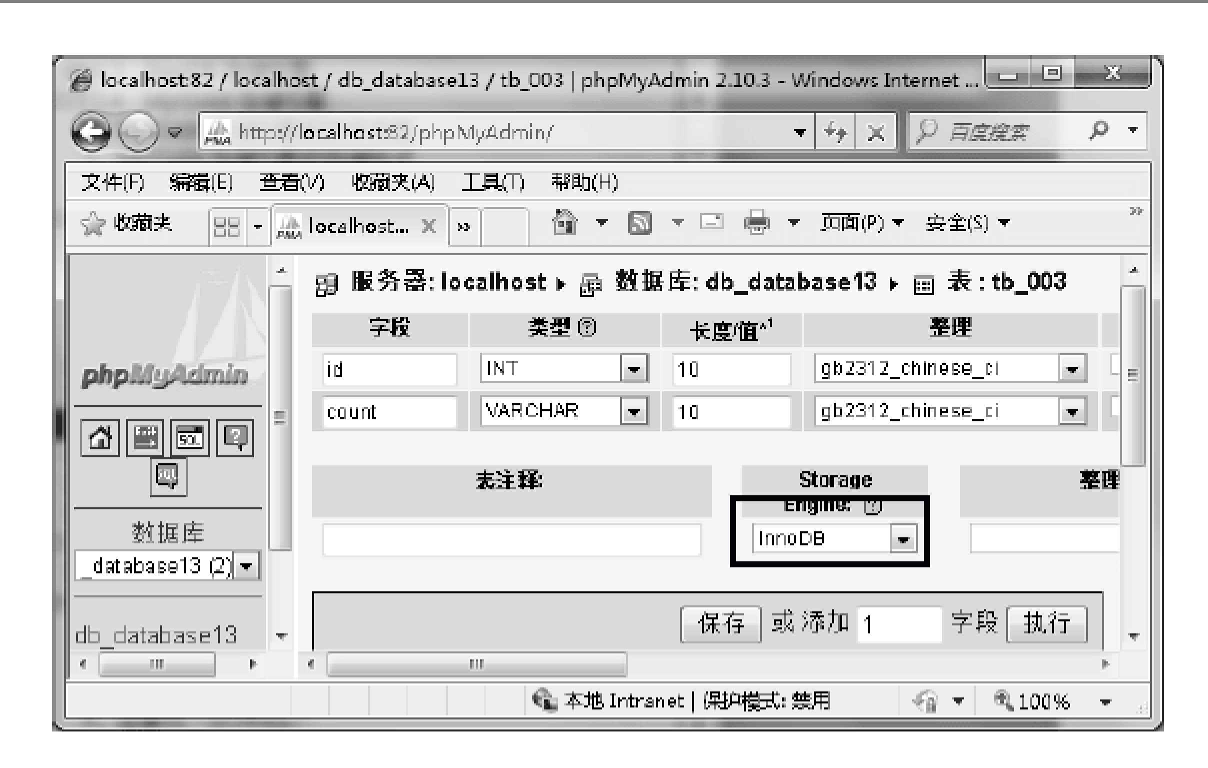


图 10.4 创建 tb_002 数据表及生成的数据表文件

（3）创建 tb_003 数据表，设置存储引擎为 InnoDB，生成的数据表文件如图 10.5 所示，同样也由一个后缀为.frm 的文件组成。



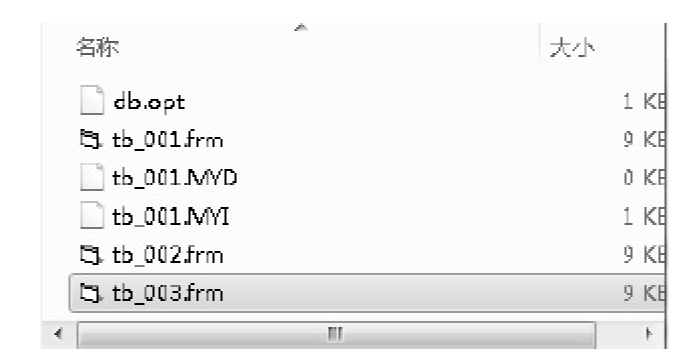


图 10.5 创建 tb_003 数据表及生成的数据表文件

10.2 MySQL 数据类型

视频讲解：光盘\TM\Video\第 10 章\MySQL 数据类型.exe

在 MySQL 数据库中，每一条数据都有其数据类型。MySQL 支持的数据类型，主要分成 3 类：数字类型、字符串（字符）类型、日期和时间类型。

10.2.1 数字类型

MySQL 支持所有的 ANSI/ISO SQL 92 数字类型。这些类型包括准确数字的数据类型（NUMERIC、DECIMAL、INTEGER 和 SMALLINT），还包括近似数字的数据类型（FLOAT、REAL 和 DOUBLE PRECISION）。其中的关键字 INT 是 INTEGER 的同义词，关键字 DEC 是 DECIMAL 的同义词。

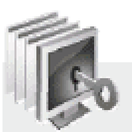
数字类型总体可以分成整型和浮点型两类，详细内容如表 10.1 和表 10.2 所示。

表 10.1 整数数据类型

数 据 类 型	取 值 范 围	说 明	单 位
TINYINT	符号值：-127～127 无符号值：0～255	最小的整数	1 字节
BIT	符号值：-127～127 无符号值：0～255	最小的整数	1 字节
BOOL	符号值：-127～127 无符号值：0～255	最小的整数	1 字节
SMALLINT	符号值：－32768～32767 无符号值：0～65535	小型整数	2 字节
MEDIUMINT	符号值：－8388608～8388607 无符号值：0～16777215	中型整数	3 字节
INT	符号值：－2147683648～2147683647 无符号值：0～4294967295	标准整数	4 字节
BIGINT	符号值： -9223372036854775808～9223372036854775807 无符号值：0～18446744073709551615	大整数	8 字节

表 10.2 浮点数据类型

数 据 类 型	取 值 范 围	说 明	单 位
FLOAT	+（-）3.402823466E+38	单精度浮点数	8 或 4 字节
DOUBLE	+（-）1.7976931348623157E+308 +（-）2.2250738585072014E-308	双精度浮点数	8 字节
DECIMAL	可变	一般整数	自定义长度



技巧 在创建表时，使用哪种数字类型，应遵循以下原则。

（1）选择最小的可用类型，如果值永远不超过 127，则使用 TINYINT 比 INT 强。

（2）对于完全都是数字的，可以选择整数类型。

（3）浮点类型用于可能具有小数部分的数，如货物单价、网上购物交付金额等。

10.2.2 字符串类型

字符串类型可以分为 3 类：普通的文本字符串类型（CHAR 和 VARCHAR）、可变类型（TEXT 和 BLOB）和特殊类型（SET 和 ENUM）。它们之间都有一定的区别，取值的范围不同，应用的地方也不同。

（1）普通的文本字符串类型，即 CHAR 和 VARCHAR 类型，CHAR 列的长度被固定为创建表所声明的长度，取值在 1～255 之间；VARCHAR 列的值是变长的字符串，取值和 CHAR 一样。下面介绍普通的文本字符串类型，如表 10.3 所示。

表 10.3 常规字符串类型

类 型	取 值 范 围	说 明
[national] char(M) [binary|ASCII|unicode]	0～255 个字符	固定长度为 M 的字符串，其中 M 的取值范围为 0～255。national 关键字指定了应该使用的默认字符集。binary 关键字指定了数据是否区分大小写（默认是区分大小写的）。ASCII 关键字指定了在改列中使用 latin1 字符集。unicode 关键字指定了使用 UCS 字符集
char	0～255 个字符	和 char(M)类似
[national] varchar(M) [binary]	0～255 个字符	长度可变，其他和 char(M)类似

（2）TEXT 和 BLOB 类型。它们的大小可以改变，TEXT 类型适合存储长文本，而 BLOB 类型适合存储二进制数据，支持任何数据，如文本、声音和图像等。下面介绍 TEXT 和 BLOB 类型，如表 10.4 所示。

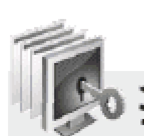
表 10.4 TEXT 和 BLOB 类型

类 型	最大长度（字节数）	说 明
TINYBLOB	2^8～1（225）	小 BLOB 字段
TINYTEXT	2^8～1（225）	小 TEXT 字段
BLOB	2^16～1（65 535）	常规 BLOB 字段
TEXT	2^16～1（65 535）	常规 TEXT 字段
MEDIUMBLOB	2^24～1（16 777 215）	中型 BLOB 字段
MEDIUMTEXT	2^24～1（16 777 215）	中型 TEXT 字段
LONGBLOB	2^32～1（4 294 967 295）	长 BLOB 字段
LONGTEXT	2^32～1（4 294 967 295）	长 TEXT 字段

（3）特殊类型 SET 和 ENUM。特殊类型 SET 和 ENUM 的介绍如表 10.5 所示。

表 10.5　ENUM 和 SET 类型

类　　型	最 大 值	说　　明
Enum ("value1", "value2", …)	65 535	该类型的列只可以容纳所列值之一或为 NULL
Set ("value1", "value2", …)	64	该类型的列可以容纳一组值或为 NULL


技巧 在创建表时，使用字符串类型时应遵循以下原则。

（1）从速度方面考虑，要选择固定的列，可以使用 CHAR 类型。

（2）要节省空间，使用动态的列，可以使用 VARCHAR 类型。

（3）要将列中的内容限制在一种选择，可以使用 ENUM 类型。

（4）允许在一个列中有多于一个的条目，可以使用 SET 类型。

（5）如果要搜索的内容不区分大小写，可以使用 TEXT 类型。

（6）如果要搜索的内容区分大小写，可以使用 BLOB 类型。

10.2.3　日期和时间数据类型

日期和时间类型包括 DATETIME、DATE、TIMESTAMP、TIME 和 YEAR。其中的每种类型都有其取值的范围，如赋予它一个不合法的值，将会被“0”代替。下面介绍日期和时间数据类型，如表 10.6 所示。

表 10.6　日期和时间数据类型

类　　型	取 值 范 围	说　　明
DATE	1000-01-01　9999-12-31	日期，格式 YYYY-MM-DD
TIME	-838:58:59　835:59:59	时间，格式 HH:MM:SS
DATETIME	1000-01-01 00:00:00 9999-12-31 23:59:59	日期和时间，格式 YYYY-MM-DD HH:MM:SS
TIMESTAMP	1970-01-01 00:00:00 2037 年的某个时间	时间标签，在处理报告时使用显示格式取决于 M 的值
YEAR	1901-2155	年份可指定两位数字和四位数字的格式

在 MySQL 中，日期的顺序是按照标准的 ANSISQL 格式进行输出的。

10.3　MySQL 运算符

数据库中的表结构确立后，表中的数据代表的意义就已经确定。而通过 MySQL 运算符进行运算，就可以获取到表结构以外的另一种数据。例如，学生表中存在一个 birth 字段，这个字段是表示学生的出生年份。而运用 MySQL 的算术运算符用当前的年份减学生出生的年份，那么得到的就是这个学生的实际年龄数据。

这就是 MySQL 的运算符，所以熟悉并掌握运算符的应用，对于操作 MySQL 数据库中的数据是非常有用的。下面就来熟悉一下 MySQL 支持的 4 种运算符都具备哪些功能。

算术运算符：执行算术运算，如加、减、乘、除等。

比较运算符：包括大于、小于、等于或者不等于等。主要用于数值的比较、字符串的匹配等方面。例

如，LIKE、IN、BETWEEN AND 和 IS NULL 等都是比较运算符，还包括正则表达式的 REGEXP 也是比较运算符。

逻辑运算符：包括与、或、非和异或等逻辑运算。其返回值为布尔型，真值（1 或 true）和假值（0 或 false）。

位运算符：包括按位与、按位或、按位取反、按位异或、按位左移和按位右移等位运算。注意位运算必须先将数据转换为二进制，然后在二进制格式下进行操作。

说明 逻辑运算符和位运算符都有与、或和异或等操作。但是，位运算符必须先把数值变成二进制类型，然后再进行按位操作。运算完成后，将二进制的值转换为原来的类型，返回给用户。逻辑运算直接进行运算，结果只返回真值（1 或 true）和假值（0 或 false）。

10.3.1 算术运算符

视频讲解：光盘\TM\Video\第 10 章\算术运算符.exe

算术运算符是 MySQL 中最常用的一类运算符。MySQL 支持的算术运算符包括加、减、乘、除和求余。下面列出算术运算符的符号、作用、表达式的形式，如表 10.7 所示。

表 10.7 算术运算符

符　　号	作　　用
+	加法运算
-	减法运算
*	乘法运算
/	除法运算
%	求余运算
DIV	除法运算，返回商。同“/”
MOD	求余运算，返回余数。同“%”

技巧 加（+）、减（-）和乘（*）可以同时运算多个操作数。除号（/）和求余运算符（%）也可以同时计算多个操作数，但是这两个符号计算多个操作数不太好。DIV 和 MOD 这两个运算符只有两个参数。进行除法和求余的运算时，如果 x2 参数是 0 时，计算结果将是空值（NULL）。

例 10.1 使用算术运算符对 tb_book1 表中 row 字段值进行加、减、乘、除运算，计算结果如图 10.6 所示。（**实例位置：光盘\TM\Instance\10\10.1**）

结果输出了 row 字段的原值，以及执行算术运算符后得到的值。

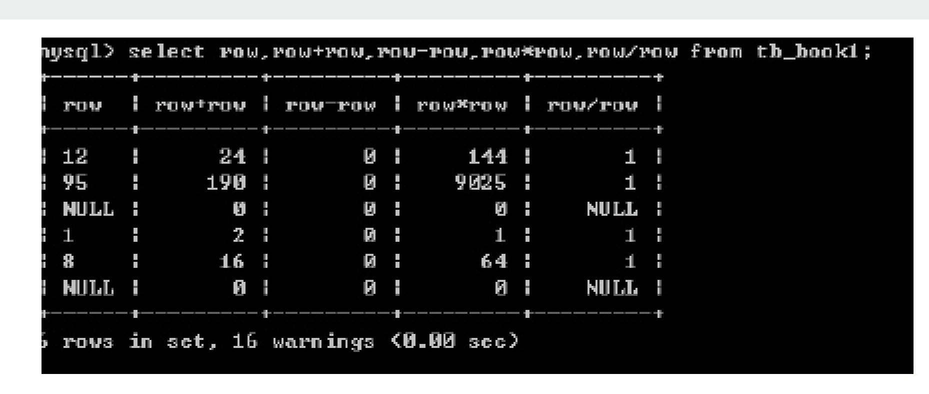


图 10.6 使用算术运算符计算数据

10.3.2 比较运算符

视频讲解：光盘\TM\Video\第 10 章\比较运算符.exe

比较运算符是查询数据时最常用的一类运算符。SELECT 语句中的条件语句经常要使用比较运算符。通



过这些比较运算符，可以判断表中的哪些记录是符合条件的。比较运算符的符号、名称和应用示例如表 10.8 所示。

表 10.8 比较运算符

运 算 符	名 称	示 例	运 算 符	名 称	示 例
=	等于	Id=5	Is not null	n/a	Id is not null
>	大于	Id>5	Between	n/a	Id between1 and 15
<	小于	Id<5	In	n/a	Id in (3,4,5)
=>	大于等于	Id=>5	Not in	n/a	Name not in (shi,li)
<=	小于等于	Id<=5	Like	模式匹配	Name like ('shi%')
!=或<>	不等于	Id!=5	Not like	模式匹配	Name not like ('shi%')
Is null	n/a	Id is null	Regexp	常规表达式	Name 正则表达式

下面对几种较常用的比较运算符进行详解。

1．运算符“=”

“=”用来判断数字、字符串和表达式等是否相等。如果相等，返回 1，否则返回 0。

说明 在运用“=”运算符判断两个字符是否相同时，数据库系统都是根据字符的 ASCII 码进行判断的。如果 ASCII 码相等，则表示这两个字符相同。如果 ASCII 码不相等，则表示两个字符不同。忌空值（NULL）不能使用“=”来判断。

例 10.2 运用“=”运算符查询出 id 等于 27 的记录，查询结果如图 10.7 所示。（**实例位置：光盘\TM\Instance\10\10.2**）

从结果中可以看出，id 等于 27 的记录返回值为 1，id 不等于 27 的记录，返回值则为 0。

2．运算符“<>”和“!=”

“<>”和“!=”用来判断数字、字符串、表达式等是否不相等。如果不相等，则返回 1；否则，返回 0。这两个符号也不能用来判断空值（NULL）。

例 10.3 运用“<>”和“!=”运算符判断 tb_book 表中 row 字段值是否等于 1、41、24。运算结果如图 10.8 所示。（**实例位置：光盘\TM\Instance\10\10.3**）

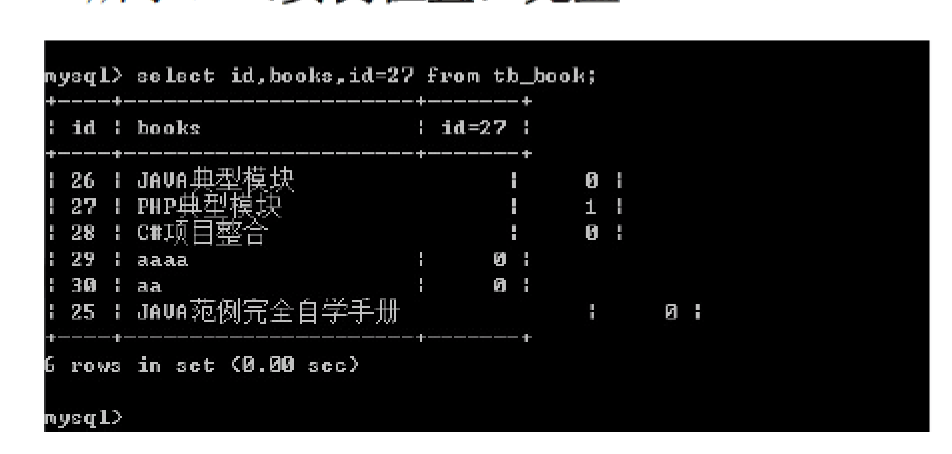


图 10.7 使用“=”查询记录

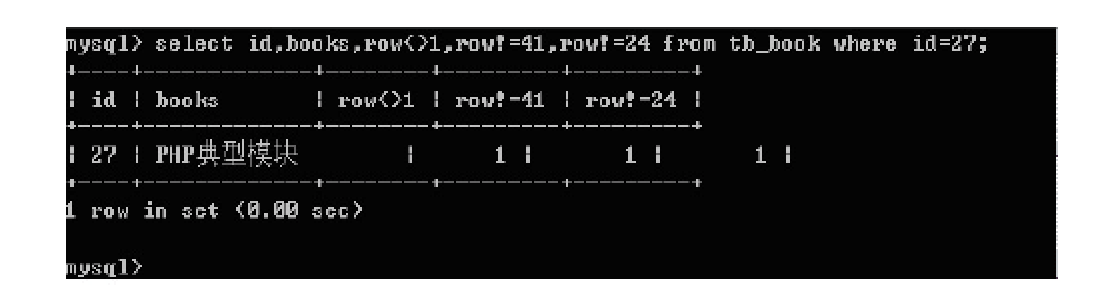


图 10.8 使用“<>”和“!=”运算符判断数据

结果显示返回值都为 1，这表示记录中的 row 字段值不等于 1、41、24。

3．运算符“>”

“>”用来判断左边的操作数是否大于右边的操作数。如果大于，返回 1；否则，返回 0。同样空值（NULL）不能使用“>”来判断。

例 10.4　使用“>”运算符来判断 tb_book 表中 row 字段值是否大于 90，是则返回 1，否则返回 0，空值返回 NULL。运算结果如图 10.9 所示。**（实例位置：光盘\TM\Instance\10\10.4）**

说明

“<”、“<=”、“>=”运算符都与“>”运算符如出一辙，其使用方法基本相同，这里不再赘述了。

4．运算符“IS NULL”

“IS NULL”用来判断操作数是否为空值（NULL）。操作数为 NULL 时，结果返回 1；否则，返回 0。IS NOT NULL 刚好与 IS NULL 相反。

例 10.5　运用“IS NULL”运算符来判断 tb_book 表中 row 字段值是否为空值，查询结果如图 10.10 所示。**（实例位置：光盘\TM\Instance\10\10.5）**

```
mysql> select id,books,row>90 from tb_book;
+----+------------------------+--------+
| id | books                  | row>90 |
+----+------------------------+--------+
| 26 | JAVA典型模块                  |      0 |
| 27 | PHP典型模块                   |      1 |
| 28 | C#项目整合                    |   NULL |
| 29 | aaaa                   |      0 |
| 30 | aa                     |      0 |
| 25 | JAVA范例完全自学手册                  |   NULL |
+----+------------------------+--------+
6 rows in set (0.00 sec)

mysql>
```



图 10.9　使用“>”运算符查询数据



图 10.10　使用“IS NULL”运算符来判断字段值是否为空

结果显示，row 字段值为空的返回值为 1，不为空的返回值为 0。

说明

“=”、“<>”、“!=”、“>”、“>=”、“<”、“<=”等运算符都不能用来判断空值（NULL）。一旦使用，结果将返回 NULL。如果要判断一个值是否为空值，可以使用“<=>”、“IS NULL”和“IS NOT NULL”来判断。注意：NULL 和‘NULL’是不同的，前者表示为空值，后者表示一个由 4 个字母组成的字符串。

5．运算符“BETWEEN AND”

“BETWEEN AND”用于判断数据是否在某个取值范围内。其表达式如下：

```
x1 BETWEEN m AND n
```

如果 x1 大于等于 m，且小于等于 n，结果将返回 1，否则将返回 0。

例 10.6　运用“BETWEEN AND”运算符判断 tb_book 表中，row 字段的值是否在 10～50 及 25～28 之间，查询结果如图 10.11 所示。**（实例位置：光盘\TM\Instance\10\10.6）**

从查询结果中可以看出，在范围内则返回 1，否则返回 0，空值返回 NULL。

6．运算符“IN”

“IN”用于判断数据是否存在于某个集合中。其表达式如下：

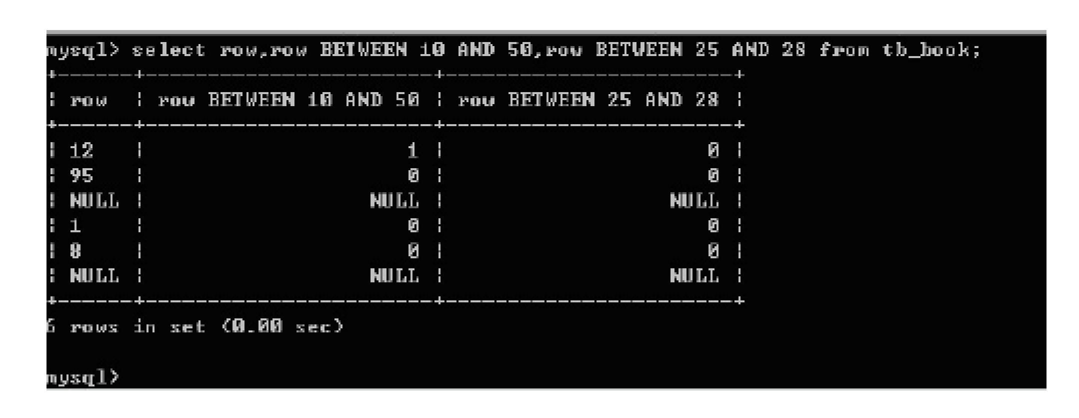
```
x1 IN(值 1,值 2,…,值 n)
```

如果 x1 等于值 1 到值 n 中的任何一个值，结果将返回 1。如果不是，结果将返回 0。

例 10.7　运用“IN”运算符判断 tb_book 表中 row 字段的值是否在指定的范围内，查询结果如图 10.12 所示。**（实例位置：光盘\TM\Instance\10\10.7）**

从查询结果可知，在范围内则返回 1，否则返回 0，空值返回 NULL。

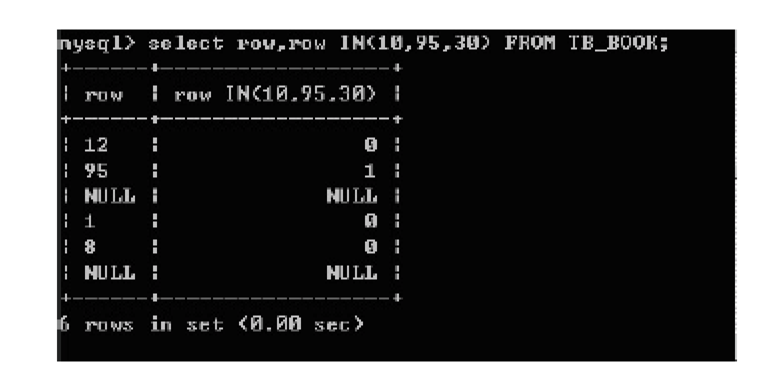




```
mysql> select row,row BETWEEN 10 AND 50,row BETWEEN 25 AND 28 from tb_book;
+------+-----------------------+-----------------------+
| row  | row BETWEEN 10 AND 50 | row BETWEEN 25 AND 28 |
+------+-----------------------+-----------------------+
| 12   |                     1 |                     0 |
| 95   |                     0 |                     0 |
| NULL |                  NULL |                  NULL |
| 1    |                     0 |                     0 |
| 8    |                     0 |                     0 |
| NULL |                  NULL |                  NULL |
+------+-----------------------+-----------------------+
6 rows in set (0.00 sec)

mysql>
```

图 10.11　使用“BETWEEN AND”运算符判断 row 字段值的范围



```
mysql> select row,row IN(10,95,30) FROM TB_BOOK;
+------+------------------+
| row  | row IN(10,95,30) |
+------+------------------+
| 12   |                0 |
| 95   |                1 |
| NULL |             NULL |
| 1    |                0 |
| 8    |                0 |
| NULL |             NULL |
+------+------------------+
6 rows in set (0.00 sec)
```

图 10.12　使用“IN”运算符判断 row 字段值的范围

7. 运算符“LIKE”

“LIKE”用来匹配字符串。其表达式如下：

```
x1 LIKE s1
```

如果 x1 与字符串 s1 匹配，结果将返回 1。否则返回 0。

例 10.8　使用“LIKE”运算符判断 tb_book 表中的 user 字段值是否与指定的字符串匹配，查询结果如图 10.13 所示。（**实例位置：光盘\TM\Instance\10\10.8**）

从查询结果可知，user 字段值为 mr 字符的记录，结果则返回 1，否则返回 0；user 字段值中包含 l 字符的记录，匹配则返回 1，否则返回 0。

8. 运算符“REGEXP”

“REGEXP”同样用于匹配字符串，但其使用的是正则表达式进行匹配。其表达式格式如下：

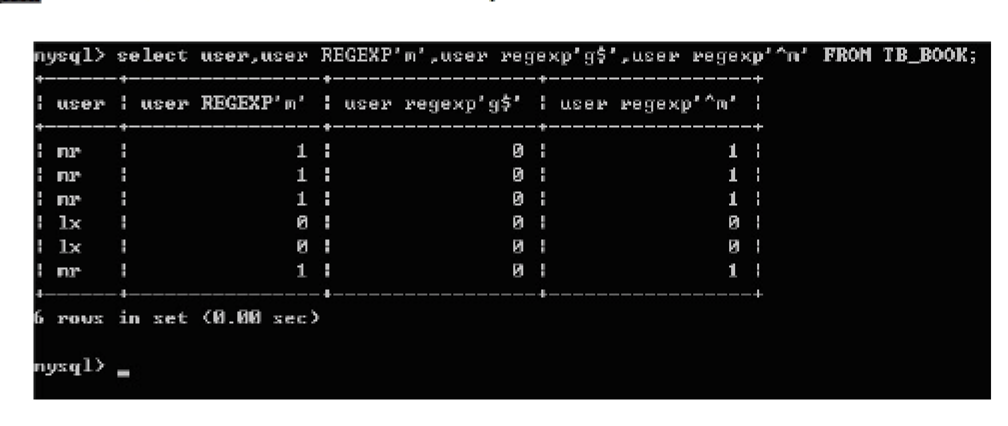
```
x1 REGEXP '匹配方式'
```

如果 x1 满足匹配方式，结果将返回 1；否则将返回 0。

例 10.9　使用“REGEXP”运算符来匹配 user 字段的值是否以指定字符开头、结尾，同时是否包含指定的字符串，执行结果如图 10.14 所示。（**实例位置：光盘\TM\Instance\10\10.9**）

```
mysql> select user,user like'mr',user like'%l%' FROM TB_BOOK;
+------+--------------+----------------+
| user | user like'mr' | user like'%l%' |
+------+--------------+----------------+
| mr   |            1 |              0 |
| mr   |            1 |              0 |
| mr   |            1 |              0 |
| lx   |            0 |              1 |
| lx   |            0 |              1 |
| mr   |            1 |              0 |
+------+--------------+----------------+
6 rows in set (0.02 sec)
```

图 10.13　使用“LIKE”运算符判断 user 字段是否匹配某字符



```
mysql> select user,user REGEXP'm',user regexp'g$',user regexp'^m' FROM TB_BOOK;
+------+----------------+-----------------+-----------------+
| user | user REGEXP'm' | user regexp'g$' | user regexp'^m' |
+------+----------------+-----------------+-----------------+
| mr   |              1 |               0 |               1 |
| mr   |              1 |               0 |               1 |
| mr   |              1 |               0 |               1 |
| lx   |              0 |               0 |               0 |
| lx   |              0 |               0 |               0 |
| mr   |              1 |               0 |               1 |
+------+----------------+-----------------+-----------------+
6 rows in set (0.00 sec)

mysql>
```

图 10.14　使用“REGEXP”运算符匹配字符串

本例使用“REGEXP”运算符判断 tb_book 表中 user 字段的值，是否以 m 字符开头；是否以 g 字符结尾；在 user 字段值中是否包含 m 字符，如果满足条件则返回 1，否则返回 0。

> **技巧**　使用“REGEXP”运算符匹配字符串，其使用方法非常简单。“REGEXP”运算符经常与“^”、“$”和“.”一起使用。“^”用来匹配字符串的开始部分；“$”用来匹配字符串的结尾部分；“.”用来代表字符串中的一个字符。

10.3.3　逻辑运算符

视频讲解：光盘\TM\Video\第 10 章\逻辑运算符.exe

逻辑运算符用来判断表达式的真假。如果表达式是真，结果返回 1。如果表达式是假，结果返回 0。逻辑运算符又称为布尔运算符。MySQL 中支持 4 种逻辑运算符，分别是与、或、非和异或。下面是 4 种逻辑运算符的符号及作用，如表 10.9 所示。

表 10.9　逻辑运算符

符　　号	作　　用
&&或 AND	与
||或 OR	或
!或 NOT	非
XOR	异或

1．与运算

“&&”或者“AND”是与运算的两种表达方式。如果所有数据不为 0 且不为空值（NULL）时，结果返回 1；如果存在任何一个数据为 0 时，结果返回 0；如果存在一个数据为 NULL 且没有数据为 0 时，结果返回 NULL。与运算符支持多个数据同时进行运算。

例 10.10　运用“&&”运算符判断 row 字段的值是否存在 0 或者 NULL（“row&&1”（row 字段值与 1）和“row&&0”（row 字段值与 0）），如果存在则返回 1，否则返回 0，空值返回 NULL。执行结果如图 10.15 所示。**（实例位置：光盘\TM\Instance\10\10.10）**

结果显示，“row&&1”中，row 字段的值为非 0，则返回 1；row 字段的值为 NULL，则返回 NULL；“row&&0”中包含 0，所以返回值为 0。

2．或运算

“||”或者“OR”表示或运算。所有数据中存在任何一个数据不为非 0 的数字时，结果返回 1；如果数据中不包含非 0 的数字，但包含 NULL 时，结果返回 NULL；如果操作数中只有 0 时，结果返回 0。或运算符“||”也可以同时操作多个数据。

例 10.11　运用“OR”运算符判断 tb_book 表中 row 是否包含 NULL 或者非 0 数字（“row OR 1”和“row OR 0”）。执行结果如图 10.16 所示。**（实例位置：光盘\TM\Instance\10\10.11）**

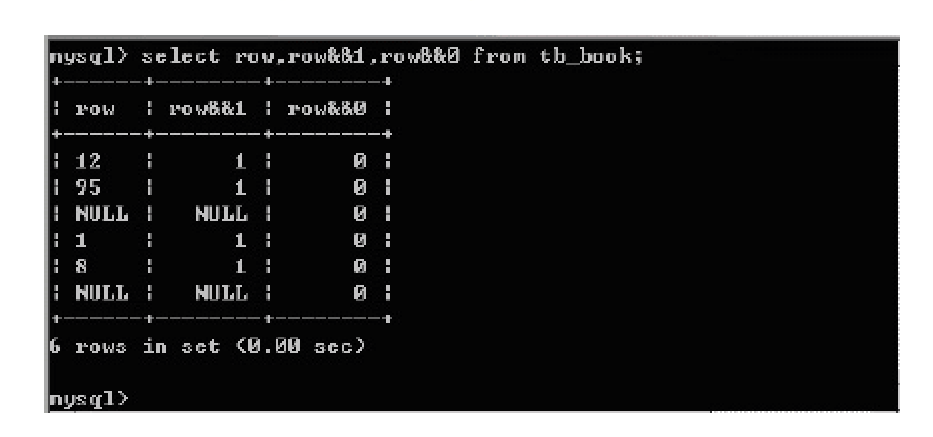


图 10.15　使用“&&”运算符判断数据

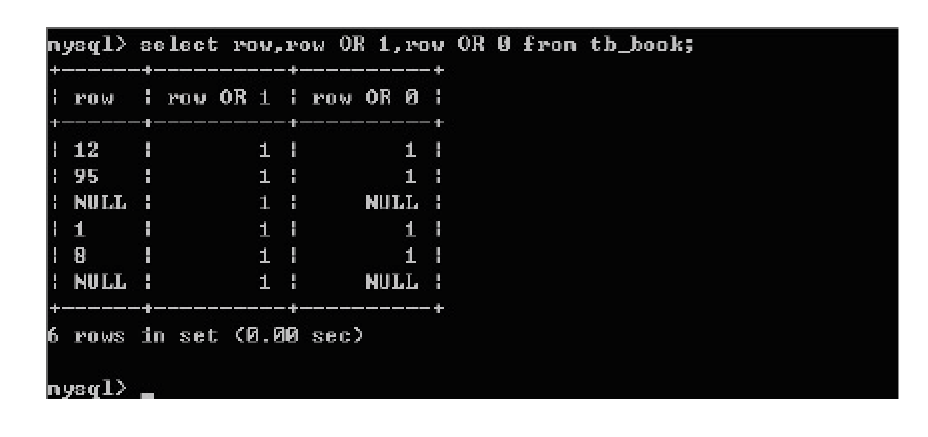


图 10.16　使用“OR”运算符匹配数据

结果显示，“row OR 1”中包含 NULL 和 1 这个非 0 的数字，所以返回结果为 1；“row OR 0”中包含非 0 的数字、NULL 和 0 的数字，所以返回 NULL 和 1。

3．非运算

“!”或者“NOT”表示非运算。通过非运算，将返回与操作数据相反的结果。如果操作数据是非 0 的数字，结果返回 0；如果操作数据是 0，结果返回 1；如果操作数据是 NULL，结果返回 NULL。

例 10.12　运用“!”运算符判断 tb_book 表中 row 字段的值是否为 0 或者 NULL。执行结果如图 10.17 所示。**（实例位置：光盘\TM\Instance\10\10.12）**

结果显示，row 字段中值为 NULL 的记录，返回值为 NULL；不为 0 的记录，返回值为 0。

4．异或运算

“XOR”表示异或运算。只要其中任何一个操作数据为 NULL 时，结果返回 NULL；如果 x1 和 x2 中一个是非 0，另一个是 0 时，结果返回 1。

例 10.13　使用“XOR”运算符判断 tb_book 表中字段 row 的值是否为 NULL（“row XOR 1”和“row



XOR 0”）。执行结果如图 10.18 所示。（**实例位置：光盘\TM\Instance\10\10.13**）

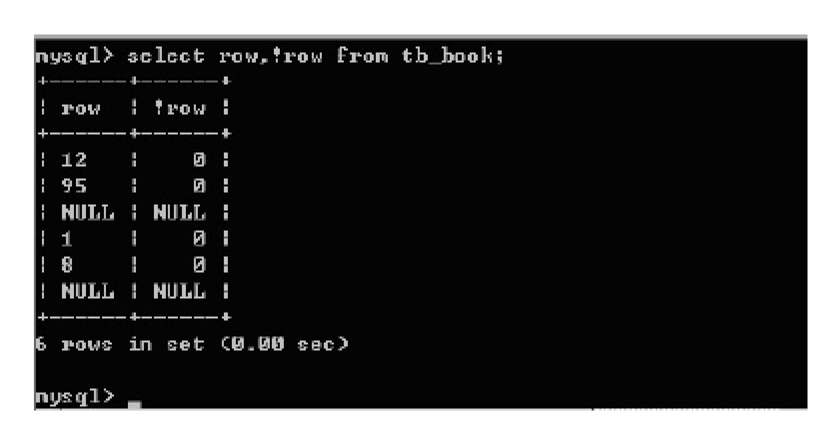



图 10.17　使用“!”运算符判断数据

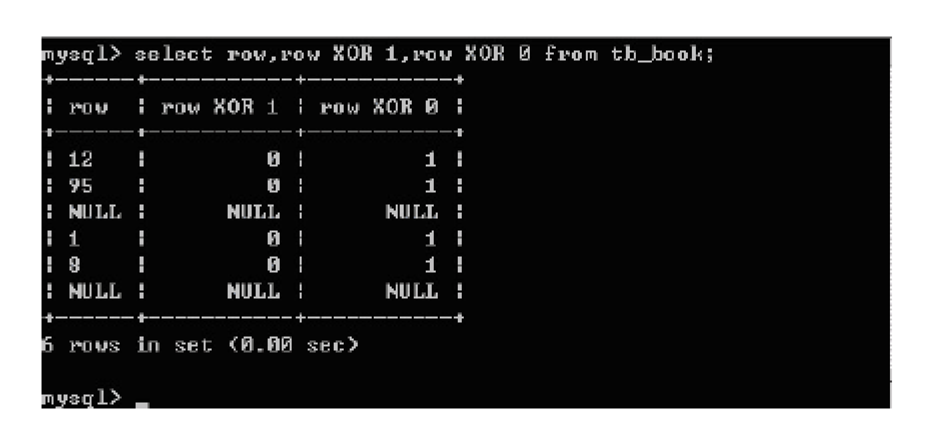


图 10.18　使用“XOR”运算符判断数据

结果显示，“row XOR 1”中 row 字段中的值为非 0 数字和 NULL 值，所以返回值为 0 和 NULL；“row XOR 0”中包含 0，所以返回值为 1，而 row 字段值为 NULL 的记录，返回值则为 NULL。

10.3.4　位运算符

位运算符是在二进制数上进行计算的运算符。位运算会先将操作数变成二进制数，进行位运算。然后再将计算结果从二进制数变回十进制数。MySQL 中支持 6 种位运算符，分别是按位与、按位或、按位取反、按位异或、按位左移和按位右移。6 种位运算符的符号及作用如表 10.10 所示。

表 10.10　位运算符

符　　号	作　　用
&	按位与。进行该运算时，数据库系统会先将十进制的数转换为二进制的数。然后对应操作数的每个二进制位上进行与运算。1 和 1 相与得 1，与 0 相与得 0。运算完成后再将二进制数变回十进制数
|	按位或。将操作数化为二进制数后，每位都进行或运算。1 和任何数进行或运算的结果都是 1，0 与 0 或运算结果为 0
~	按位取反。将操作数化为二进制数后，每位都进行取反运算。1 取反后变成 0，0 取反后变成 1
^	按位异或。将操作数化为二进制数后，每位都进行异或运算。相同的数异或后结果是 0，不同的数异或后结果为 1
<<	按位左移。“m<<n”表示 m 的二进制数向左移 n 位，右边补上 n 个 0。例如，二进制数 001 左移 1 位后将变成 0010
>>	按位右移。“m>>n”表示 m 的二进制数向右移 n 位，左边补上 n 个 0。例如，二进制数 011 右移 1 位后变成 001，最后一个 1 直接被移出

10.3.5　运算符的优先级

由于在实际应用中可能需要同时使用多个运算符。这就必须考虑运算符的运算顺序。正所谓：“闻道有先后，术业有专攻。”

本小节将具体阐述 MySQL 运算符使用的优先级，如表 10.11 所示。按照从高到低、从左到右的级别进行运算操作。如果优先级相同，则表达式左边的运算符先运算。

表 10.11　MySQL 运算符的优先级

优　先　级	运　算　符
1	!
2	~

续表

优　先　级	运　算　符
3	^
4	*,/,DIV,%,MOD
5	+,-
6	>>,<<
7	&
8	|
9	=,<=>,<,<=,>,>=,!=,<>,IN,IS,NULL,LIKE,REGEXP
10	BETWEEN AND,CASE,WHEN,THEN,ELSE
11	NOT
12	&&,AND
13	||,OR,XOR
14	:=

10.4　实　　战

视频讲解：光盘\TM\Video\第 10 章\实战.exe

10.4.1　查看存储引擎和创建数据库

例 10.14　登录数据库系统后，创建 student 数据库和 teacher 数据库。都创建成功后，然后查看数据库系统中还存在哪些数据库，并且查看该数据库系统支持的存储引擎类型。（**实例位置：光盘\TM\Instance\10\10.14**）

代码如下：

```
Create database student;                    //创建数据库
Create database teacher;
Show databases;                             //查看数据库
Show engines;                               //查看存储引擎
```

运行结果如图 10.19 所示。

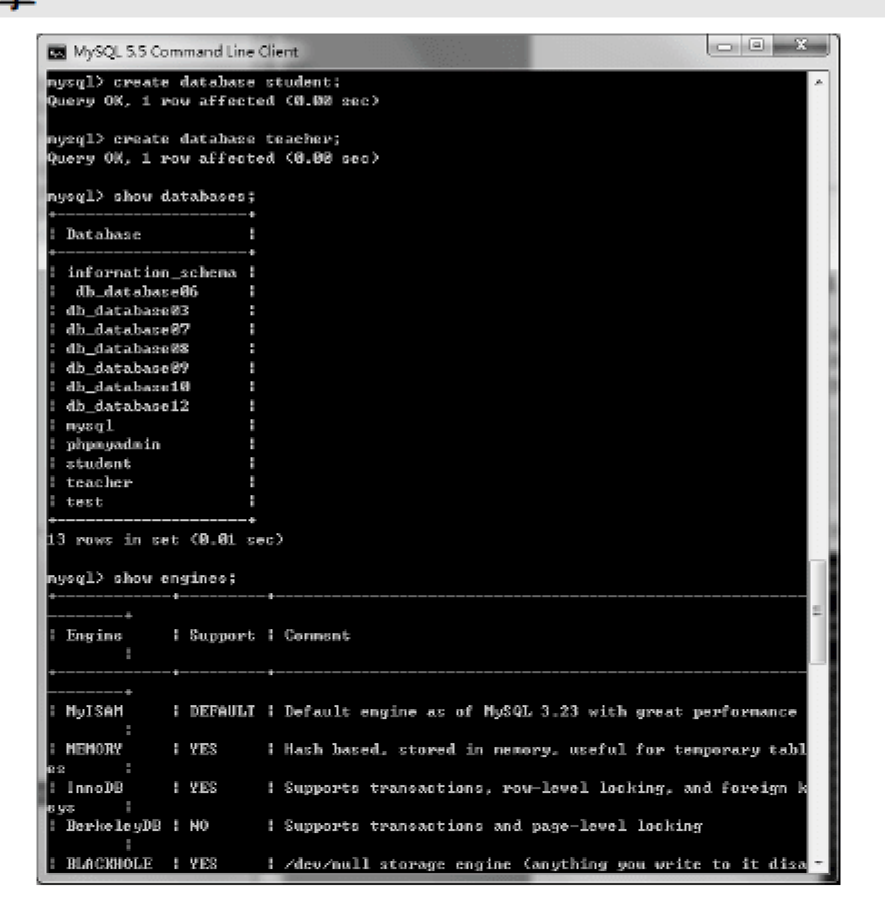


图 10.19　查询存储引擎和创建数据库

10.4.2　位运算的比较

例 10.15　本实例是将数字 4 和 6 进行按位与、按位或，并将 4 按位取反。位运算符是在二进制数上进行计算的运算符。位运算会先将操作数变成二进制数，进行位运算。然后再将计算结果从二进制数变回十进制数。（**实例位置：光盘\TM\Instance\10\10.15**）

代码如下：

```
mysql>select 4&6,4|6,~4;
```

运行结果如图 10.20 所示。

10.4.3　逻辑运算的使用

例 10.16　将数字 2、0 和 null 之间的任意两个进行逻辑运算，逻辑运算符用来判断表达式的真假。如果表达式是真，结果返回 1；如果表达式是假，结果返回 0。逻辑运算符又称为布尔运算符。**（实例位置：光盘\TM\Instance\10\10.16）**

关键代码参考如下：

```
mysql>select 2&&0,2&&null,0 and null,2||0,2||null,0 or null;
```

运算结果如图 10.21 所示。

10.4.4　浮点数类型

例 10.17　某表的字段 a、b 和 c 的数据类型分别是 FLOAT、DOUBLE 和 DECIMAL，向表中插入 3.143、3.145、3.1434，然后查询该表的数据。**（实例位置：光盘\TM\Instance\10\10.17）**

代码如下：

```
Select * from aa;
```

运行结果如图 10.22 所示。

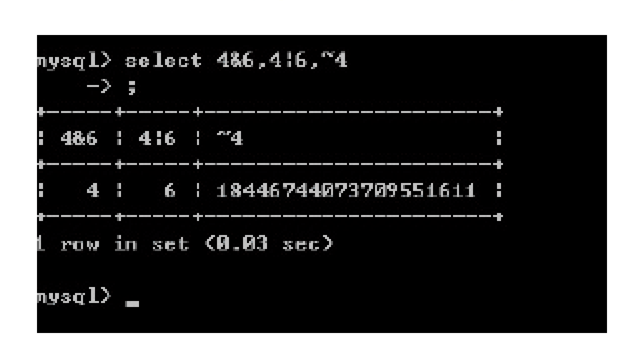


图 10.20　位运算的比较

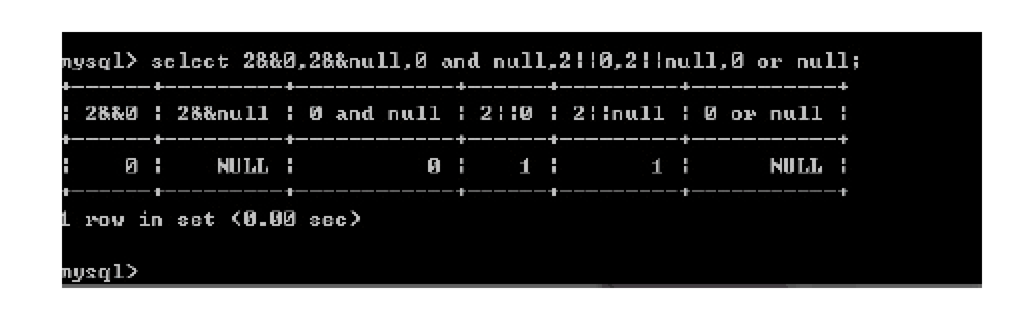


图 10.21　逻辑运算的使用

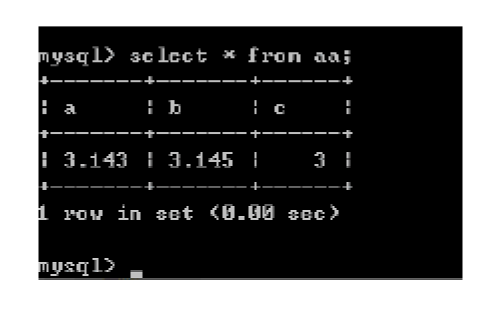


图 10.22　数据类型的显示

10.5　小　　结

本章对 MySQL 存储引擎、数据类型、运算符进行了详细讲解，并通过举例说明，使读者更好地理解所学知识的用法。在阅读本章时，读者应该重点掌握什么类型的表适合什么类型的存储引擎，数据类型及各种运算符的使用，同时对 MySQL 中的数据类型也要有一定的了解。读者需要认真学习这部分的内容，位运算符是本章的难点。因为，位运算符需要将操作数转换为二进制数，然后进行位运算。这要求读者能够掌握二进制运算的相关知识。

10.6　学习成果检验

1．使用算术运算符计算 5*6/6 的值。**（实例位置：光盘\TM\Instance\10\10.18）**

2．在 MySQL 中执行下面的比较运算符的表达式：40>=30，40<=30，NULL<=>NULL，7<=>7。**（实例位置：光盘\TM\Instance\10\10.19）**

3．在 MySQL 中执行下列位运算：11&15，11|15，13^15，～15。**（实例位置：光盘\TM\Instance\10\10.20）**

4．MySQL 中什么数据类型能够储存路径？**（实例位置：光盘\TM\Instance\10\10.21）**

第 11 章

MySQL 函数之选

（视频讲解：26 分钟）

MySQL 数据库中提供了很丰富的函数。MySQL 函数包括数学函数、字符串函数、日期和时间函数、条件判断函数、系统信息函数、加密函数、格式化函数等。通过这些函数，可以简化用户的操作。例如，字符串连接函数可以很方便地将多个字符串连接在一起。

通过阅读本章内容，你可以：

- 了解 MySQL 函数
- 了解 MySQL 数学函数的使用方法
- 掌握 MySQL 字符串函数的使用方法
- 掌握 MySQL 日期和时间函数的使用
- 掌握条件判断函数的应用
- 掌握系统函数和加密函数的使用
- 掌握其他函数的使用

11.1　MySQL 函数

MySQL 函数是 MySQL 数据库提供的内置函数。这些内置函数可以帮助用户更加方便地处理表中的数据。本节中将简单地介绍 MySQL 中包含哪些类别的函数，以及这些函数的使用范围和作用。MySQL 中内置的函数及其作用如表 11.1 所示。

表 11.1　MySQL 内置函数类别及作用

函　数	作　用
数学函数	用于处理数字。这类函数包括绝对值函数、正弦函数、余弦函数和获取随机数函数等
字符串函数	用于处理字符串。其中包括字符串连接函数、字符串比较函数、字符串中字母大小写转换函数等
日期和时间函数	用于处理日期和时间。其中包括获取当前时间的函数、获取当前日期的函数、返回年份的函数和返回日期的函数等
条件判断函数	用于在 SQL 语句中控制条件选择。其中包括 IF 语句、CASE 语句和 WHEN 语句等
系统信息函数	用于获取 MySQL 数据库的系统信息。其中包括获取数据库名的函数、获取当前用户的函数和获取数据库版本的函数等
加密函数	用于对字符串进行加密解密。其中包括字符串加密函数和字符串解密函数等
其他函数	包括格式化函数和锁函数等

MySQL 的内置函数不但可以在 SELECT 查询语句中应用，同样也可以在 INSERT、UPDATE 和 DELETE 等语句中应用。例如，在 INSERT 添加语句中，应用日期时间函数获取系统的当前时间，并且将其添加到数据表中。MySQL 内置函数可以对表中数据进行相应地处理，以便得到用户希望得到的数据。有了这些内置函数可以使 MySQL 数据库的功能更加强大。下面将对 MySQL 的内置函数逐一进行详细介绍。

11.2　数 学 函 数

视频讲解：光盘\TM\Video\第 11 章\数学函数.exe

数学函数是 MySQL 中常用的一类函数。其主要用于处理数字，包括整型和浮点数等。MySQL 中内置的数学函数及其作用如表 11.2 所示。

表 11.2　MySQL 的数学函数

函　数	作　用
ABS(x)	返回 x 的绝对值
CEIL(x),CEILIN(x)	返回不小于 x 的最小整数值
FLOOR(x)	返回不大于 x 的最大整数值
RAND()	返回 0～1 的随机数
RAND(x)	返回 0～1 的随机数，x 值相同时返回的随机数相同
SIGN(x)	返回参数作为-1、0 或 1 的符号，该符号取决于 x 的值为负、零或正
PI()	返回圆周率的值。默认的显示小数位数是 7 位，然而 MySQL 内部会使用完全双精度值
TRUNCATE(x,y)	返回数值 x 保留到小数点后 y 位的值
ROUND(x)	返回离 x 最近的整数

续表

函　　数	作　　用
ROUND(x,y)	保留 x 小数点后 y 位的值，但截断时要进行四舍五入
POW(x,y),POWER(x,y)	返回 x 的 y 乘方的结果值
SQRT(x)	返回非负数 x 的二次方根
EXP(x)	返回 e 的 x 乘方后的值（自然对数的底）
MOD(x,y)	返回 x 除以 y 后的余数
LOG(x)	返回 x 的基数为 2 的对数
LOG10(x)	返回 x 的基数为 10 的对数
RADIANS(x)	将角度转换为弧度
DEGREES(x)	返回参数 x，该参数由弧度被转化为度
SIN(x)	返回 x 正弦，其中 x 在弧度中被给定
ASIN(x)	返回 x 的反正弦，即正弦为 x 的值。若 x 不在-1 到 1 的范围之内，则返回 NULL
COS(x)	返回 x 的余弦，其中 x 在弧度上已知
ACOS(x)	返回 x 反余弦，即余弦是 x 的值。若 x 不在-1 到 1 的范围之内，则返回 NULL
TAN(x)	返回 x 的反正切，即正切为 x 的值
ATAN(x),ATAN2(x,y)	返回两个变量 x 及 y 的反正切。它类似于 y 或 x 的反正切计算，除非两个参数的符号均用于确定结果所在象限
COT(x)	返回 x 的余切

下面对其中的常用函数进行讲解，并且配合以实例做详细说明。

11.2.1　ABS(x)函数

ABS(x)函数用于求绝对值。

例 11.1　使用 ABS(x)函数来求 8 和-8 的绝对值。**（实例位置：光盘\TM\Instance\11\11.1）**

其语句如下：

```
select ABS(8),ABS(-8);
```

查询结果如图 11.1 所示。

11.2.2　FLOOR(x)函数

FLOOR(x)函数返回小于或等于 x 的最大整数。

例 11.2　应用 FLOOR(x)函数求出 3.6 和-4 的最大整数。**（实例位置：光盘\TM\Instance\11\11.2）**

其语句如下：

```
select FLOOR(3.6),FLOOR(-4);
```

其查询结果如图 11.2 所示。



图 11.1　使用 ABS(x)求绝对值

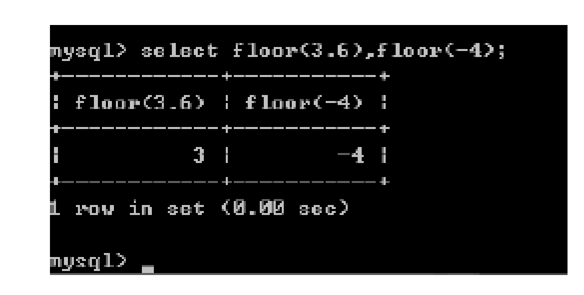


图 11.2　使用 FLOOR(x)函数求出最大整数



11.2.3　RAND()函数

RAND()函数是返回 0～1 的随机数。但是 RAND()返回的数是完全随机的。

例 11.3　运用 RAND()函数，获取两个随机数。**（实例位置：光盘\TM\Instance\11\11.3）**

其语句如下：

```
Select RAND(),RAND();
```

其查询结果如图 11.3 所示。

11.2.4　PI()函数

PI()函数用于返回圆周率。

例 11.4　使用 PI()函数获取圆周率。**（实例位置：光盘\TM\Instance\11\11.4）**

其语句如下：

```
select PI();
```

查询结果如图 11.4 所示。

11.2.5　TRUNCATE(x,y)函数

TRUNCATE(x,y)函数返回 x 保留到小数点后 y 位的值。

例 11.5　使用 TRUNCATE(x,y)函数返回 3.987654 小数点后 3 位的值。**（实例位置：光盘\TM\Instance\11\11.5）**

其语句如下：

```
select TRUNCATE(3.987654,3);
```

其查询结果如图 11.5 所示。

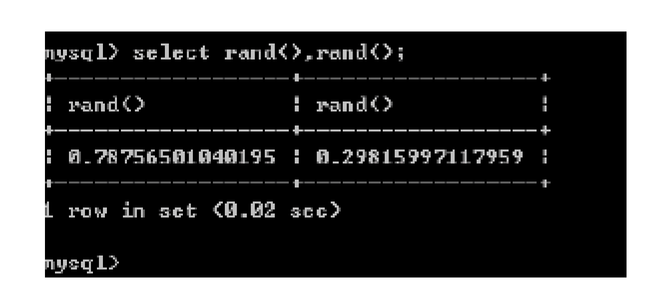


图 11.3　使用 RAND()函数获取随机数

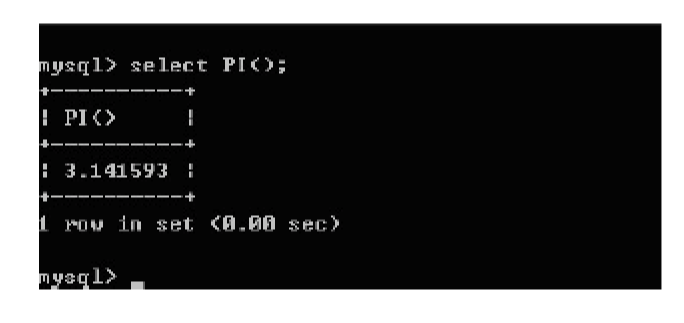


图 11.4　使用 PI()函数获取圆周率

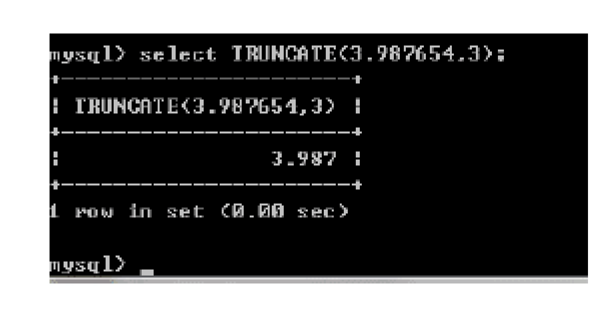


图 11.5　使用 TRUNCATE(x,y)函数获取数据

11.2.6　ROUND(x)函数和 ROUND(x,y)函数

ROUND(x)函数返回离 x 最近的整数，也就是对 x 进行四舍五入处理；ROUND(x,y)函数返回 x 保留到小数点后 y 位的值，截断时需要进行四舍五入处理。

例 11.6　使用 ROUND(x)函数获取 2.7 和 1.8 最近的整数，使用 ROUND(x,y)函数获取 3.123456 小数点后 3 位的值。**（实例位置：光盘\TM\Instance\11\11.6）**

其语句如下：

```
select ROUND(2.7),ROUND(1.8),ROUND(3.123456,3);
```

查询结果如图 11.6 所示。

11.2.7 SQRT(x)函数

SQRT(x)函数用于求平方根。

例 11.7 使用 SQRT(x)函数求 36 和 25 的平方根。（**实例位置：光盘\TM\Instance\11\11.7**）

其语句如下：

```
select SQRT(36),SQRT(25);
```

其查询结果如图 11.7 所示。

```
mysql> select ROUND(2.7),ROUND(1.8),ROUND(3.123456,3);
+------------+------------+-------------------+
| ROUND(2.7) | ROUND(1.8) | ROUND(3.123456,3) |
+------------+------------+-------------------+
|          3 |          2 |             3.123 |
+------------+------------+-------------------+
1 row in set (0.00 sec)

mysql>
```

图 11.6 使用 ROUND(x)函数和 ROUND(x,y)函数获取数据

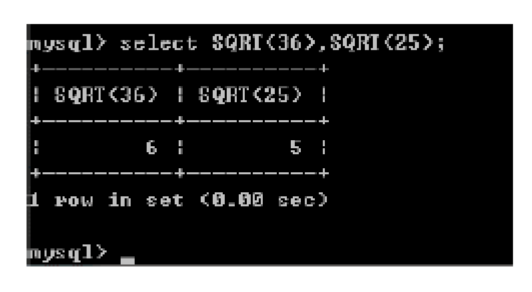


图 11.7 使用 SQRT(x)函数求 16 和 25 的平方根

11.3 字符串函数

视频讲解：光盘\TM\Video\第 11 章\字符串函数.exe

字符串函数是 MySQL 中最常用的一类函数。字符串函数主要用于处理表中的字符串。MySQL 内置的字符串函数及其作用如表 11.3 所示。

表 11.3 MySQL 的字符串函数

函　　数	作　　用
CHAR_LENGTH(s)	返回字符串 s 的字符数
LENGTH(s)	返回值为字符串 s 的长度，单位为字节。一个多字节字符算作多字节。这意味着对于一个包含 5 个两字节字符的字符串，LENGTH()的返回值为 10，而 CHAR_LENGTH()的返回值则为 5
CONCAT(s1,s2,…)	返回结果为连接参数产生的字符串。如有任何一个参数为 NULL，则返回值为 NULL。或许有一个或多个参数。如果所有参数均为非二进制字符串，则结果为非二进制字符串。如果自变量中含有任一二进制字符串，则结果为一个二进制字符串。一个数字参数被转化为与之相等的二进制字符串格式；若要避免这种情况，可使用显式类型 cast，如 SELECT CONCAT(CAST(int_col AS CHAR), char_col)
CONCAT_WS(x,s1,s2,…)	同 CONCAT(s1,s2,…)函数，但是每个字符串直接要加上 x
INSERT(s1,x,len,s2)	将字符串 s2 替换 s1 的 x 位置开始长度为 len 的字符串
UPPER(s),UCASE(s)	将字符串 s 的所有字母都变成大写字母
LOWER(s),LCASE(s)	将字符串 s 的所有字母都变成小写字母
LEFT(s,n)	返回从字符串 s 开始的最左边 n 个字符
RIGHT(s,n)	返回从字符串 s 开始的最右边 n 个字符
LPAD(s1,len,s2)	返回字符串 s1，其左边由字符串 s2 填补到 len 字符长度。假如 s1 的长度大于 len，则返回值被缩短至 len 字符
RPAD(s1,len,s2)	返回字符串 s1，其右边被字符串 s2 填补至 len 字符长度。假如字符串 s1 的长度大于 len，则返回值被缩短到与 len 字符相同长度

续表

函　　数	作　　用
LTRIM(s)	返回字符串 s，其引导空格字符被删除
RTRIM(s)	返回字符串 s，结尾空格字符被删去
TRIM(s)	去掉字符串 s 开始处和结尾处的空格
TRIM(s1 FROM s)	去掉字符串 s 中开始处和结尾处的字符串 s1
REPEAT(s,n)	将字符串 s 重复 n 次
SPACE(n)	返回 n 个空格
REPLACE(s,s1,s2)	用字符串 s2 替代字符串 s 中的字符串 s1
STRCMP(s1,s2)	比较字符串 s1 和 s2
SUBSTRING(s,n,len)	获取从字符串 s 第 n 个位置开始长度为 len 的字符串
MID(s,n,len)	同 SUBSTRING(s,n,len)
LOCATE(s1,s),POSITION (s1 IN s)	从字符串 s 中获取 s1 的开始位置
INSTR(s,s1)	从字符串 s 中获取 s1 的开始位置
REVERSE(s)	将字符串 s 的顺序反过来
ELT(n,s1,s2,…)	返回第 n 个字符串
EXPORT_SET(bits,on,off[, separator[,number_of_bits]])	返回一个字符串，生成规则如下：针对 bits 的二进制格式，如果其位为 1，则返回一个 on 值；如果其位为 0，则返回一个 off 值。每个字符串使用 separator 进行分隔，默认值为“,”。number_of_bits 参数指定 bits 可用的位数，默认值为 64 位。例如，生成数字 182 的二进制（10110110）替换格式，以“@”作为分隔符，设置有效位为 6 位。其语句如下：select EXPORT_SET(182,'Y','N','@',6); 其运行结果为：N@Y@Y@N@Y@Y
FIELD(s,s1,s2,…)	返回第一个与字符串 s 匹配的字符串的位置
FIND_IN_SET(s1,s2)	返回在字符串 s2 中与 s1 匹配的字符串的位置
MAKE_SET(x,s1,s2,…)	按 x 的二进制数从 s1,s2,…,sn 中选取字符串

下面对其中的常用函数进行讲解，并且配合以例子做详细说明。

11.3.1　INSERT(s1,x,len,s2)函数

INSERT(s1,x,len,s2)函数将字符串 s1 中 x 位置开始长度为 len 的字符串用字符串 s2 替换。

例 11.8　使用 INSERT()函数将 mrkj 字符串中的 kj 替换为 book。**（实例位置：光盘\TM\Instance\11\11.8）**

其语句如下：

```
select INSERT('mrkj',3,2,'book');
```

替换后的查询结果如图 11.8 所示。

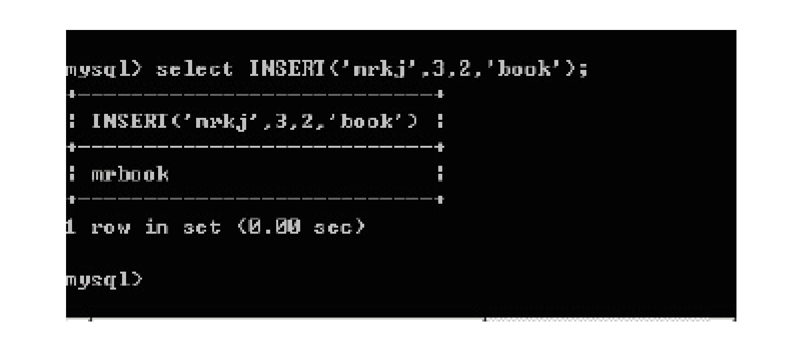


图 11.8　使用 INSERT()函数替换指定字符串

11.3.2　UPPER(s)、UCASE(s)函数

UPPER(s)函数和 UCASE(s)函数将字符串 s 的所有字母变成大写字母。

例 11.9　下面使用 UPPER(s)函数和 UCASE(s)函数将 mingri 字符串中的所有字母变成大写字母。**（实例位置：光盘\TM\Instance\11\11.9）**

其语句如下：

```
select UPPER('mingri'),UCASE('mingri');
```

其转换后的结果如图 11.9 所示。

11.3.3 LEFT(s,n)函数

LEFT(s,n)函数返回字符串 s 的前 n 个字符。

例 11.10 应用 LEFT()函数返回 mingribook 字符串的前 6 个字符。（**实例位置：光盘\TM\Instance\11\11.10**）

其语句如下：

```
select LEFT('mingribook',6);
```

其截取结果如图 11.10 所示。

```
mysql> select UPPER('mingri'),UCASE('mingri');
+-----------------+-----------------+
| UPPER('mingri') | UCASE('mingri') |
+-----------------+-----------------+
| MINGRI          | MINGRI          |
+-----------------+-----------------+
1 row in set (0.04 sec)

mysql>
```

图 11.9 使用 UPPER(s)函数和 UCASE(s)函数将 mingri 字符串中的所有字母变成大写字母



图 11.10 使用 LEFT 函数返回指定字符

11.3.4 RTRIM(s)函数

RTRIM(s)函数将去掉字符串 s 结尾处的空格。

例 11.11 应用 RTRIM()函数去掉 mingri 结尾处的空格。（**实例位置：光盘\TM\Instance\11\11.11**）

其语句如下：

```
select CONCAT('+',RTRIM(' mingri '),'+');
```

其结果如图 11.11 所示。

11.3.5 SUBSTRING(s,n,len)函数

SUBSTRING(s,n,len)函数从字符串 s 的第 n 个位置开始获取长度为 len 的字符串。

例 11.12 下面使用 SUBSTRING()函数获取从 mingribook 字符串的第 1 位开始的 6 个字符，结果如图 11.12 所示。（**实例位置：光盘\TM\Instance\11\11.12**）

```
mysql> select CONCAT('+',RTRIM(' mingri '),'+');
+-----------------------------------+
| CONCAT('+',RTRIM(' mingri '),'+') |
+-----------------------------------+
| + mingri+                         |
+-----------------------------------+
1 row in set (0.00 sec)

mysql>
```

图 11.11 使用 RTRIM 函数去掉 mingri 结尾处的空格

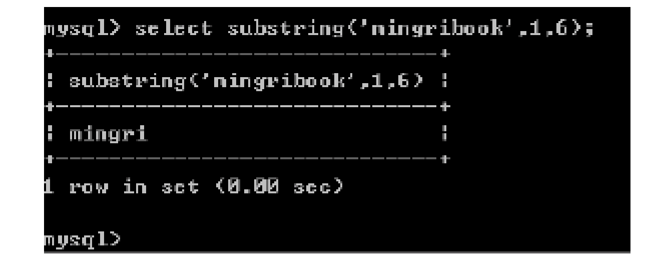


图 11.12 使用 SUBSTRING 函数获取指定长度字符串

11.3.6 REVERSE(s)函数

REVERSE(s)函数将字符串 s 的顺序反过来。

例 11.13 下面使用 REVERSE()函数将 mingri 字符串的顺序反过来，结果如图 11.13 所示。（**实例位**



置：光盘\TM\Instance\11\11.13）

11.3.7　FIELD(s,s1,s2,…)函数

FIELD(s,s1,s2,…)函数返回第一个与字符串 s 匹配的字符串的位置。

例 11.14　应用 FIELD()函数返回第一个与字符串 mr 匹配的字符串位置，结果如图 11.14 所示。**（实例位置：光盘\TM\Instance\11\11.14）**



图 11.13　使用 REVERSE()函数将 mingri 字符串的顺序反过来

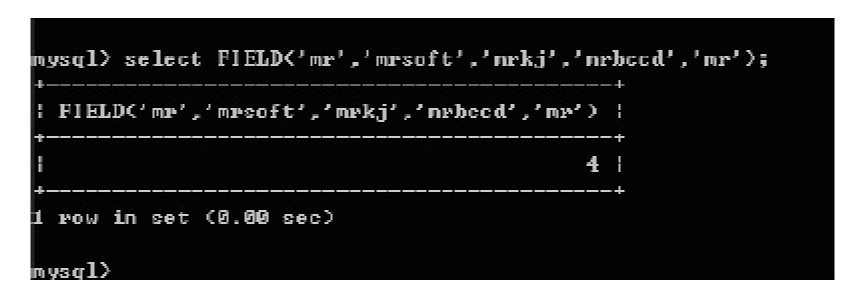


图 11.14　使用 FIELD()函数返回第一个与字符串 mr 匹配的字符串位置

11.4　日期和时间函数

视频讲解：光盘\TM\Video\第 11 章\日期和时间函数.exe

日期和时间函数是 MySQL 中另一最常用的函数。其主要用于对表中的日期和时间数据的处理。MySQL 内置的日期时间函数及作用如表 11.4 所示。

表 11.4　MySQL 的日期和时间函数

函　　数	作　　用
CURDATE(),CURRENT_DATE()	返回当前日期
CURTIME(),CURRENT_TIME()	返回当前时间
NOW(),CURRENT_TIMESTAMP(),LOCALTIME(),SYSDATE(),LOCALTIMESTAMP()	返回当前日期和时间
UNIX_TIMESTAMP()	以 UNIX 时间戳的形式返回当前时间
UNIX_TIMESTAMP(d)	将时间 d 以 UNIX 时间戳的形式返回
FROM_UNIXTIME(d)	把 UNIX 时间戳的时间转换为普通格式的时间
UTC_DATE()	返回 UTC（Universal Coordinated Time，国际协调时间）日期
UTC_TIME()	返回 UTC 时间
MONTH(d)	返回日期 d 中的月份值，范围是 1～12
MONTHNAME(d)	返回日期 d 中的月份名称，如 January、February 等
DAYNAME(d)	返回日期 d 是星期几，如 Monday、Tuesday 等
DAYOFWEEK(d)	返回日期 d 是星期几，1 表示星期日，2 表示星期一等
WEEKDAY(d)	返回日期 d 是星期几，0 表示星期一，1 表示星期二等
WEEK(d)	计算日期 d 是本年的第几个星期，范围是 0～53
WEEKOFYEAR(d)	计算日期 d 是本年的第几个星期，范围是 1～53
DAYOFYEAR(d)	计算日期 d 是本年的第几天
DAYOFMONTH(d)	计算日期 d 是本月的第几天

续表

函　数	作　用
YEAR(d)	返回日期 d 中的年份值
QUARTER(d)	返回日期 d 是第几季度，范围是 1～4
HOUR(t)	返回时间 t 中的小时值
MINUTE(t)	返回时间 t 中的分钟值
SECOND(t)	返回时间 t 中的秒钟值
EXTRACT(type FROM d)	从日期 d 中获取指定的值，type 指定返回的值，如 YEAR、HOUR 等
TIME_TO_SEC(t)	将时间 t 转换为秒
SEC_TO_TIME(s)	将以秒为单位的时间 s 转换为时分秒的格式
TO_DAYS(d)	计算日期 d～0000 年 1 月 1 日的天数
FROM_DAYS(n)	计算从 0000 年 1 月 1 日开始 n 天后的日期
DATEDIFF(d1,d2)	计算日期 d1～d2 之间相隔的天数
ADDDATE(d,n)	计算起始日期 d 加上 n 天的日期
ADDDATE(d,INTERVAL expr type)	计算起始日期 d 加上一个时间段后的日期
DATE_ADD(d,INTERVAL expr type)	同 ADDDATE(d,INTERVAL n type)
SUBDATE(d,n)	计算起始日期 d 减去 n 天后的日期
SUBDATE(d,INTERVAL expr type)	计算起始日期 d 减去一个时间段后的日期
ADDTIME(t,n)	计算起始时间 t 加上 n 秒的时间
SUBTIME(t,n)	计算起始时间 t 减去 n 秒的时间
DATE_FROMAT(d,f)	按照表达式 f 的要求显示日期 d
TIME_FROMAT(t,f)	按照表达式 f 的要求显示时间 t
GET_FORMAT(type,s)	根据字符串 s 获取 type 类型数据的显示格式

下面对其中的常用函数进行讲解，并且配合以实例做详细说明。

11.4.1　CURDATE()函数和 CURRENT_DATE()函数

CURDATE()和 CURRENT_DATE()函数获取当前日期。

例 11.15　下面使用 CURDATE()和 CURRENT_DATE()函数获取当前日期。**（实例位置：光盘\TM\Instance\11\11.15）**

其语句如下：

```
select CURDATE(),CURRENT_DATE();
```

其查询结果如图 11.15 所示。

11.4.2　CURTIME()函数和 CURRENT_TIME()函数

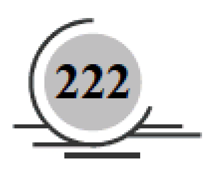
CURTIME()和 CURRENT_TIME()函数获取当前时间。

例 11.16　下面使用 CURTIME()和 CURRENT_TIME()函数获取当前时间。**（实例位置：光盘\TM\Instance\11\11.16）**

其语句如下：

```
select CURTIME(),CURRENT_TIME();
```

其查询结果如图 11.16 所示。



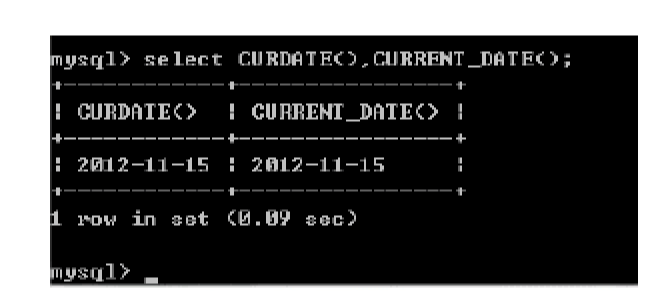


图 11.15　使用 CURDATE()和 CURRENT_DATE()函数获取当前日期

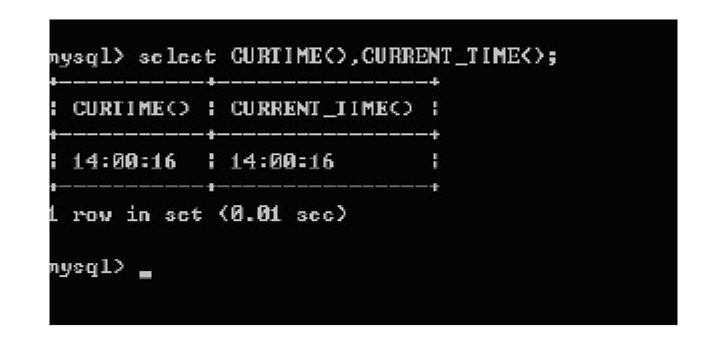


图 11.16　使用 CURTIME()和 CURRENT_TIME()函数获取当前时间

11.4.3　NOW()函数

NOW()函数获取当前日期和时间。还有 CURRENT_TIMESTAMP()、LOCALTIME()、SYSDATE()和 LOCALTIMESTAMP()函数也同样可以获取当前日期和时间。

例 11.17　下面使用 NOW()、CURRENT_TIMESTAMP()、LOCALTIME()、SYSDATE()函数来获取当前日期和时间。（**实例位置：光盘\TM\Instance\11\11.17**）

其语句如下：

```
select NOW(),CURRENT_TIMESTAMP(),LOCALTIME(),SYSDATE();
```

运行结果如图 11.17 所示。

11.4.4　DATEDIFF(d1,d2)函数

DATEDIFF(d1,d2)函数用于计算日期 d1 与 d2 之间相隔的天数。

例 11.18　使用 DATEDIFF(d1,d2)函数计算 2012-11-18 与 2018-11-11 之间相隔的天数。（**实例位置：光盘\TM\Instance\11\11.18**）

其语句如下：

```
select DATEDIFF('2012-11-18','2012-11-11');
```

结果如图 11.18 所示。

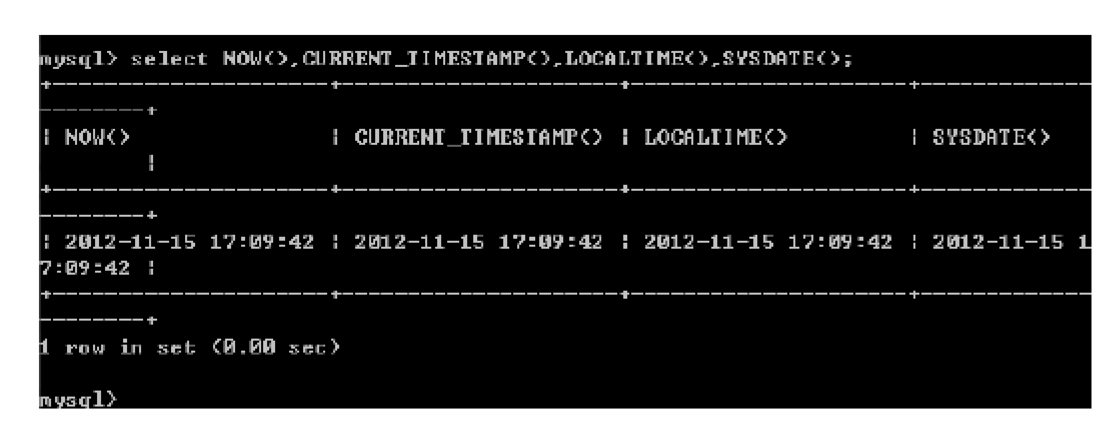


图 11.17　使用 NOW()、URRENT_TIMESTAMP()等函数获取当前日期和时间

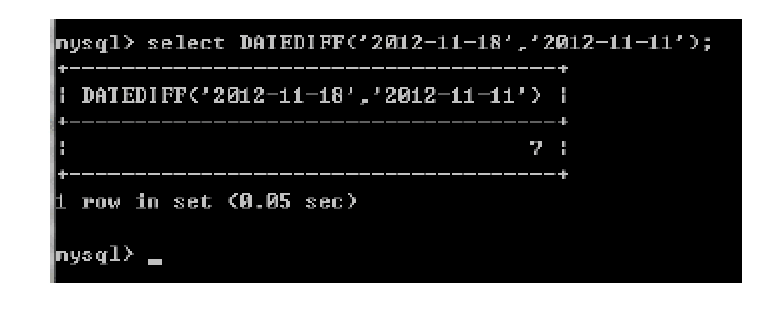


图 11.18　使用 DATEDIFF(d1,d2)函数计算 2012-11-18 与 2012-11-11 之间相隔的天数

11.4.5　ADDDATE(d,n)函数

ADDDATE(d,n)函数用于返回起始日期 d 加上 n 天的日期。

例 11.19　使用 ADDDATE(d,n)函数返回 2012-11-16 加上 2 天的日期，结果如图 11.19 所示。（**实例位置：光盘\TM\Instance\11\11.19**）

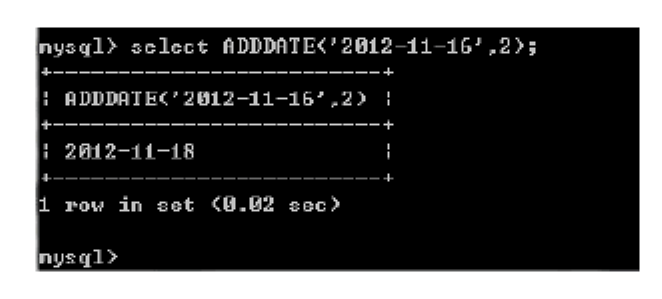


图 11.19　使用 ADDDATE(d,n)函数返回 2012-11-16 加上 2 天的日期

11.4.6 ADDDATE(d,INTERVAL expr type)函数

ADDDATE(d,INTERVAL expr type)函数返回起始日期 d 加上一个时间段后的日期。

例 11.20 使用 ADDDATE(d,INTERVAL expr type)函数返回 2012-11-16 加上 1 年 2 个月后的日期。（**实例位置：光盘\TM\Instance\11\11.20**）

其语句如下：

```
select ADDDATE('2012-11-16',INTERVAL '1 2' YEAR_MONTH);
```

其运行结果如图 11.20 所示。

11.4.7 SUBDATE(d,n)函数

SUBDATE(d,n)函数返回起始日期 d 减去 n 天的日期。

例 11.21 使用 SUBDATE(d,n)函数返回 2012-11-16 减去 4 天后的日期，结果如图 11.21 所示。（**实例位置：光盘\TM\Instance\11\11.21**）

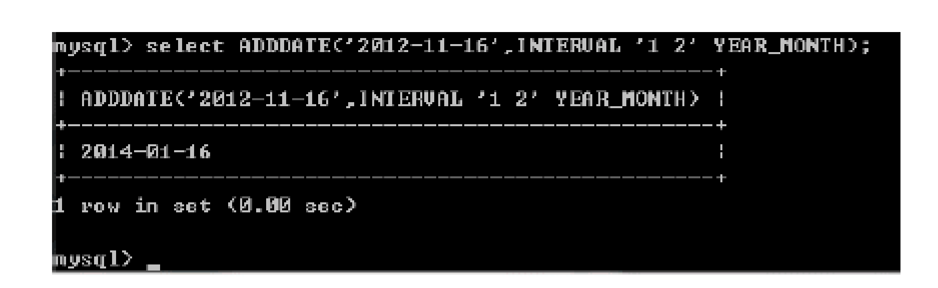


图 11.20 使用 ADDDATE(d,INTERVAL expr type)函数返回 2012-11-16 加上 1 年 2 个月后的日期

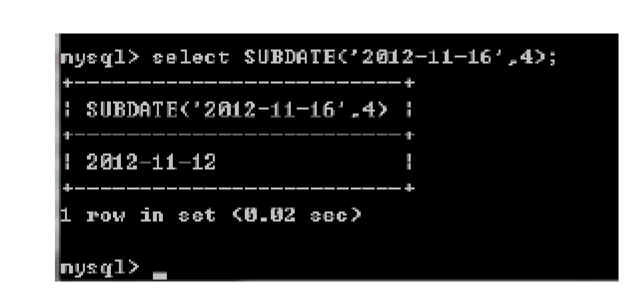


图 11.21 使用 SUBDATE(d,n)函数返回 2012-11-16 减去 4 天后的日期

11.5 条件判断函数

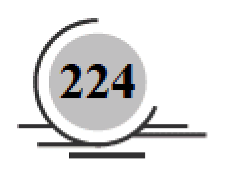
视频讲解：光盘\TM\Video\第 11 章\条件判断函数.exe

条件函数用来在 SQL 语句中进行条件判断。根据不同的条件，执行不同的 SQL 语句。MySQL 支持的条件判断函数及作用如表 11.5 所示。

表 11.5 MySQL 的条件判断函数

函　数	作　用
IF(expr,v1,v2)	如果表达式 expr 成立，则执行 v1；否则执行 v2
IFNULL(v1,v2)	如果 v1 不为空，则显示 v1 的值；否则显示 v2 的值
CASE WHEN expr1 THEN v1 [WHEN expr2 THEN v2 …][ELSE vn] END	case 表示函数开始，end 表示函数结束。如果表达式 expr1 成立，则返回 v1 的值；如果表达式 expr2 成立，则返回 v2 的值。依次类推，最后遇到 else 时，返回 vn 的值。它的功能与 PHP 中的 switch 语句类似
CASE expr WHEN e1 THEN v1 [WHEN e2 THEN v2 …][ELSE vn] END	case 表示函数开始，end 表示函数结束。如果表达式 expr 取值为 e1，则返回 v1 的值；如果表达式 expr 取值为 e2，则返回 v2 的值，依次类推，最后遇到 else，则返回 vn 的值

例 11.22 查询编程词典业绩信息表，如果业绩超过 100 万，则输出“Very Good”，如果业绩小于 100 万大于 10 万，则输出“Popularly”，否则输出“Not Good”。（**实例位置：光盘\TM\Instance\11\11.22**）



其语句如下：

```
select id,grade, CASE WHEN grade>1000000 THEN 'Very Good' WHEN grade<1000000 and grade >=100000
THEN 'Popularly' ELSE 'Not Good' END level from tb_bccd;
```

其查询结果如图 11.22 所示。

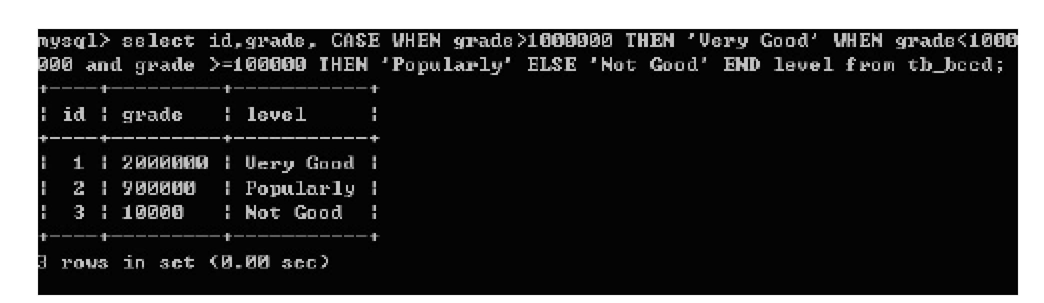


图 11.22　条件判断函数的应用

11.6　系统信息函数

系统信息函数用来查询 MySQL 数据库的系统信息。例如，查询数据库的版本、查询数据库的当前用户等。下面是各种系统信息函数的作用，如表 11.6 所示。

表 11.6　MySQL 的系统信息函数

函　　数	作　　用	示　　例
VERSION()	获取数据库的版本号	select VERSION();
CONNECTION_ID()	获取服务器的连接数	select CONNECTION_ID();
DATABASE(),SCHEMA()	获取当前数据库名	select DATABASE(),SCHEMA();
USER(),SYSTEM_USER(),SESSION_USER()	获取当前用户	select USER(),SYSTEM_USER();
CURRENT_USER(),CURRENT_USER	获取当前用户	select CURRENT_USER();
CHARSET(str)	获取字符串 str 的字符集	select CHARSET('mrsoft');
COLLATION(str)	获取字符串 str 的字符排列方式	select COLLATION('mrsoft');
LAST_INSERT_ID()	获取最近生成的 AUTO_INCREMENT 值	select LAST_INSERT_ID();

11.6.1　获取 MySQL 版本号、连接数和数据库名的函数

VERSION()函数返回数据库的版本号；CONNECTION_ID()函数返回服务器的连接数，也就是到现在为止 MySQL 服务的连接次数；DATABASE()和 SCHEMA()返回当前数据库名。

例 11.23　下面将演示 VERSION()、CONNECTION_ID()、DATABASE()和 SCHEMA() 4 个函数的用法。**（实例位置：光盘\TM\Instance\11\11.23）**

其中，VERSION()函数返回的版本号为“5.5.12”；CONNECTOIN_ID()返回的连接数为 30；DATABASE()和 SCHEMA()返回的当前数据库名是 test。

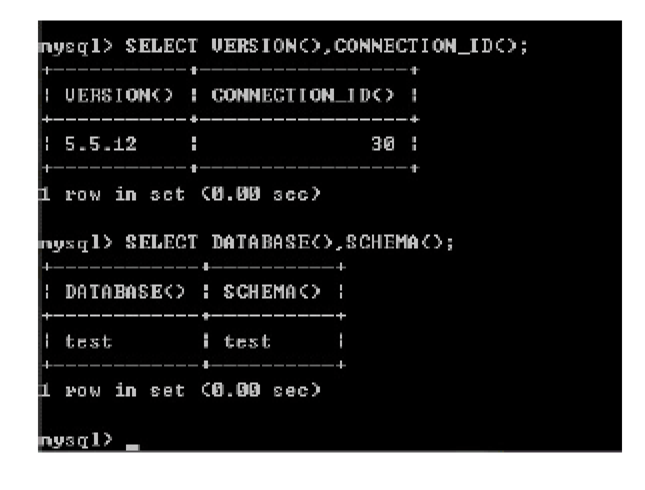


图 11.23　获取 MySQL 版本号、连接数和数据库名得函数

11.6.2　获取用户名的函数

USER()、SYSTEM_USER()、SESSION_USER()、CURRENT_USER()和 CURRENT_USER 这几个函数可以返回当前用户的名称。

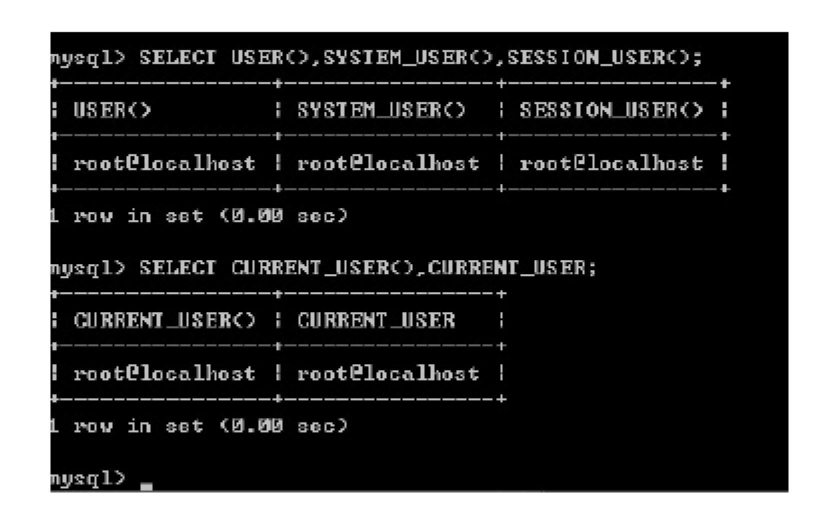
例 11.24　下面查询当前用户的用户名。运行结果如图 11.24 所示。**（实例位置：光盘\TM\Instance\11\11.24）**

结果显示，当前用户的用户名为 root。Localhost 是主机名。因为服务器和客户端在一台机器上，所以服务器的主机名为 localhost。用户名和主机名之间用符号“@”进行连接。

11.6.3 获取字符串的字符集和排序方式的函数

CHARSET(str)函数返回字符串 str 的字符集，一般情况下这个字符集就是系统的默认字符集；COLLATION(str)函数返回字符串 str 的字符排列方式。

例 11.25 下面查看字符串‘aa’的字符集和字符串排序方式。运行结果如图 11.25 所示。（**实例位置：光盘\TM\Instance\11\11.25**）

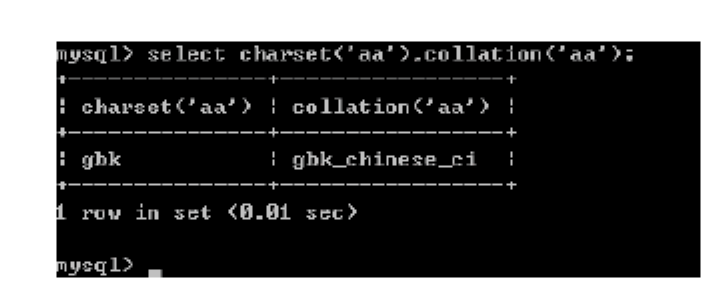


```
mysql> SELECT USER(),SYSTEM_USER(),SESSION_USER();
+----------------+----------------+----------------+
| USER()         | SYSTEM_USER()  | SESSION_USER() |
+----------------+----------------+----------------+
| root@localhost | root@localhost | root@localhost |
+----------------+----------------+----------------+
1 row in set (0.00 sec)

mysql> SELECT CURRENT_USER(),CURRENT_USER;
+----------------+----------------+
| CURRENT_USER() | CURRENT_USER   |
+----------------+----------------+
| root@localhost | root@localhost |
+----------------+----------------+
1 row in set (0.00 sec)

mysql> _
```

图 11.24 获取用户名的函数



```
mysql> select charset('aa'),collation('aa');
+---------------+-----------------+
| charset('aa') | collation('aa') |
+---------------+-----------------+
| gbk           | gbk_chinese_ci  |
+---------------+-----------------+
1 row in set (0.01 sec)

mysql> _
```

图 11.25 获取字符串的字符集和排序方式的函数

11.7 加 密 函 数

加密函数是 MySQL 中用来对数据进行加密的函数。因为数据库中有些很敏感的信息不希望被其他人看到，所以就可以通过加密的方式来使这些数据变成看似乱码的数据。例如，用户密码就应该进行加密。各种加密函数的作用如表 11.7 所示。

表 11.7 MySQL 的加密函数

函　数	作　用	示　例
PASSWORD(str)	对字符串 str 进行加密。经此函数加密后的数据是不可逆的。其经常用于对用户注册的密码进行加密处理	对字符串 mrsoft 进行密码，其语句如下： select PASSWORD('mrsoft');
MD5(str)	对字符串 str 进行加密。经常用于对普通数据进行加密	使用 MD5()函数对 mrsoft 字符串进行加密，其语句如下： select MD5('mrsoft');
ENCODE(str,pswd_str)	使用字符串 pswd_str 来加密字符串 str。加密的结果是一个二进制数，必须使用 BLOB 类型的字段来保存它	使用字符串 mr 对 mrsoft 进行加密处理，其语句如下： select ENCODE('mrsoft','mr');
DECODE(crypt_str, pswd_str)	使用字符串 pswd_str 来为 crypt_str 解密。crypt_str 是通过 ENCODE(str,pswd_str)加密后的二进制数据。字符串 pswd_str 应该与加密时的字符串 pswd_str 相同	应用 DECODE 函数对经过 ENCODE 函数加密的字符串进行解密，其语句如下： select DECODE(ENCODE('mrsoft','mr'),'mr');

11.7.1　加密函数 PASSWORD(str)

PASSWORD(str)函数可以对字符串 str 进行加密。一般情况下，PASSWORD(str)函数主要是用来给用户的密码加密的。

例 11.26　下面使用 PASSWORD(str)函数为字符串‘abcd’加密。运行结果如图 11.26 所示。（**实例位置：光盘\TM\Instance\11\11.26**）

结果显示，字符串‘abcd’加密后的结果是*A154C2565E9E7F94BFC08A1FE702624ED8EFDA。PASSWORD(str)函数加密是不可逆的。

技巧 PASSWORD(str)函数经常用来给密码加密。MySQL 用户需要设置密码，用户不能将未加密的密码直接存储到 MySQL 的 user 表中。因为登录 MySQL 数据库时，数据库系统会将你输入的密码先通过 PASSWORD(str)函数加密，然后与数据库中的密码进行比较，匹配成功后才可以登录。

11.7.2　加密函数 MD5(str)

MD5(str)函数可以对字符串 str 进行加密。主要对普通的数据进行加密。

例 11.27　下面使用 MD5(str)函数为字符串‘abcd’加密。运行结果如图 11.27 所示。（**实例位置：光盘\TM\Instance\11\11.27**）



图 11.26　加密函数 PASSWORD(str)

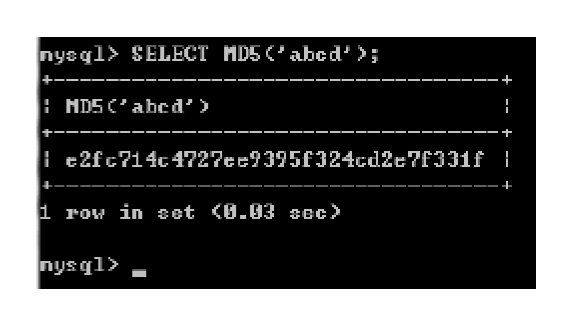


图 11.27　加密函数 MD5(str)

结果显示，字符串 abcd 的 MD5 值为 e2fc714c4727ee9395f324cd2e7f331f。

11.8　其 他 函 数

MySQL 中除了上述内置函数以外，还包含很多函数。例如，数字格式化函数 FORMAT(x,n)，IP 地址与数字的转换函数 INET_ATON(ip)，还有加锁函数 GET_LOCT(name,time)、解锁函数 RELEASE_LOCK(name)等。在表 11.8 中罗列了 MySQL 中支持的其他函数。

表 11.8　MySQL 的其他函数

函　数	作　用
FORMAT(x,n)	将数字 x 进行格式化，将 x 保留到小数点后 n 位。这个过程需要进行四舍五入
ASCII(s)	ASCII(s)返回字符串 s 的第一个字符的 ASCII 码
BIN(x)	BIN(x)返回 x 的二进制编码
HEX(x)	HEX(x)返回 x 的十六进制编码

续表

函　数	作　用
OCT(x)	OCT(x)返回 x 的八进制编码
CONV(x,f1,f2)	CONV(x,f1,f2)将 x 从 f1 进制数变成 f2 进制数
INET_ATON(IP)	INET_ATON(IP)函数可以将 IP 地址转换为数字表示
INET_NTOA(N)	INET_NTOA(N)函数可以将数字 n 转换成 IP 的形式
GET_LOCT(name,time)	GET_LOCT(name,time)函数定义一个名称为name、持续时间长度为time秒的锁。锁定成功，返回 1；如果尝试超时，返回 0；如果遇到错误，返回 NULL
RELEASE_LOCK(name)	RELEASE_LOCK(name)函数解除名称为 name 的锁。如果解锁成功，返回 1；如果尝试超时，返回 0；如果解锁失败，返回 NULL
IS_FREE_LOCK(name)	IS_FREE_LOCK(name)函数判断是否使用名为 name 的锁。如果使用，返回 0；否则返回 1
BENCHMARK(count,expr)	将表达式 expr 重复执行 count 次，然后返回执行时间。该函数可以用来判断 MySQL 处理表达式的速度
CONVERT(s USING cs)	将字符串 s 的字符集变成 cs
CAST(x AS type), CONVERT(x, type)	将 x 变成 type 类型，这两个函数只对 BINARY、CHAR、DATE、DATETIME、TIME、SIGNED INTEGER、UNSIGNED INTEGER 这些类型起作用。但两种方法只是改变了输出值的数据类型，并没有改变表中字段的类型

11.8.1　格式化函数 FORMAT(x,n)

FORMAT(x,n)函数可以将数字 x 进行格式化，将 x 保留到小数点后 n 位。这个过程需要进行四舍五入。例如，FORMAT(2.356,2)返回的结果将会是 2.36；FORMAT(2.353,2)返回的结果将会是 2.35。

例 11.28　下面使用 FORMAT(x，n)函数来将 789.12545 和 876.4321 进行格式化，都保留到小数点 3 位。运行结果如图 11.28 所示。（**实例位置：光盘\TM\Instance\11\11.28**）

结果显示，789.12545 格式化后的结果是 789.125；876.4321 格式化后的结果是 876.432。这个数都保留到小数点后 3 位，而且都进行了四舍五入处理。

注意 FORMAT(x,n)函数可以将 x 保留到小数点后 n 位。在格式化过程中需要进行四舍五入的操作。FORMAT(x,n)函数与 ROUND(x,y)函数返回 x 保留到小数点后 y 位的值。截断时需要进行四舍五入处理。

11.8.2　改变字符集的函数

CONVERT(s USING cs)函数将字符串 s 的字符集变成 cs。

例 11.29　下面将字符串‘abcd’的字符集变成 gbk。运行结果如图 11.29 所示。（**实例位置：光盘\TM\Instance\11\11.29**）

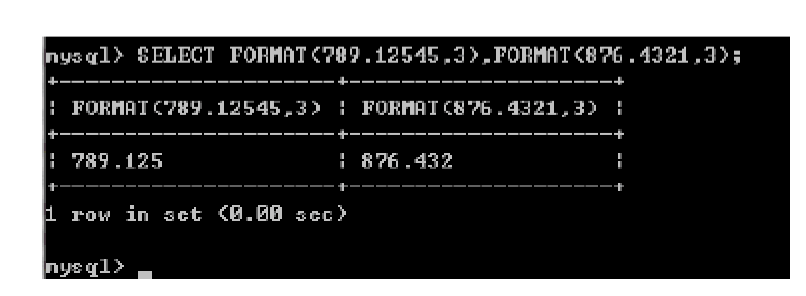


图 11.28　格式化函数 FORMAT()

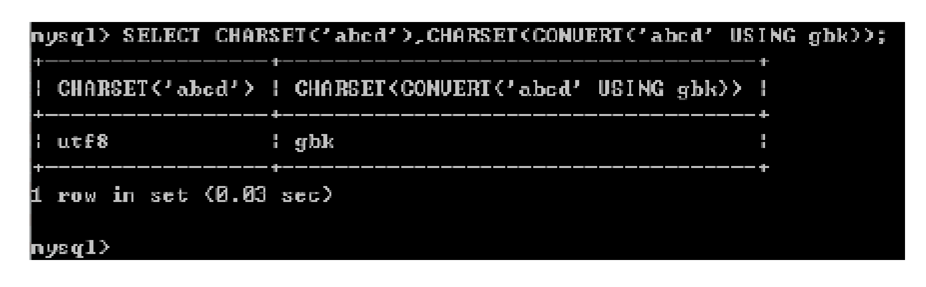
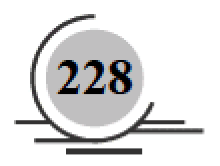


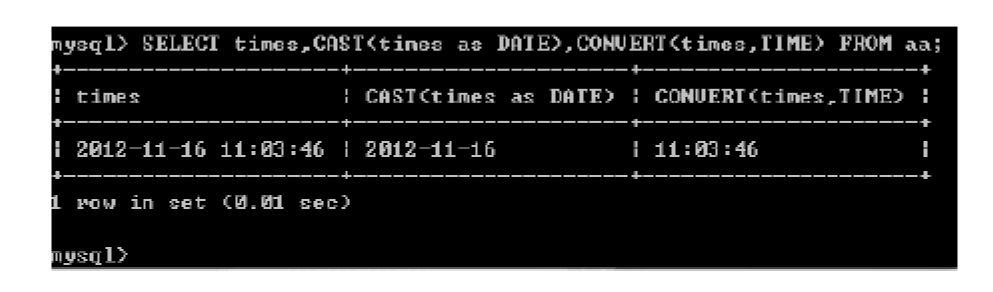
图 11.29　改变字符集的函数



11.8.3　改变字段数据类型的函数

CAST(x AS type)和 CONVERT(x,type)这两个函数将 x 变成 type 类型。这两个函数只对 BINARY、CHAR、DATETIME、TIME、SIGNED INTEGER、UNSIGNED INTEGER 这些类型起作用。但两种方法只是改变了输出值得数据类型，并没有改变表中字段的类型。

例 11.30　下面表中的 times 字段为 DATETIME 类型，将其变为 DATE 类型，或者 TIME 类型。运行结果如图 11.30 所示。**（实例位置：光盘\TM\Instance\11\11.30）**


```
mysql> SELECT times,CAST(times as DATE),CONVERT(times,TIME) FROM aa;
+---------------------+---------------------+---------------------+
| times               | CAST(times as DATE) | CONVERT(times,TIME) |
+---------------------+---------------------+---------------------+
| 2012-11-16 11:03:46 | 2012-11-16          | 11:03:46            |
+---------------------+---------------------+---------------------+
1 row in set (0.01 sec)

mysql>
```

图 11.30　改变字段数据类型的函数

结果显示，times 字段原来的取值是 2012-11-16 11:03:46，这是 DATETIME 类型；CAST(times AS DATE)返回的结果是 2012-11-16，这说明类型已经变成了 DATE 型；CONVERT(times，TIME)返回的结果是 11:03:46，这说明类型已经变成了 TIME 类型。

11.9　实　　战

视频讲解：光盘\TM\Video\第 11 章\实战.exe

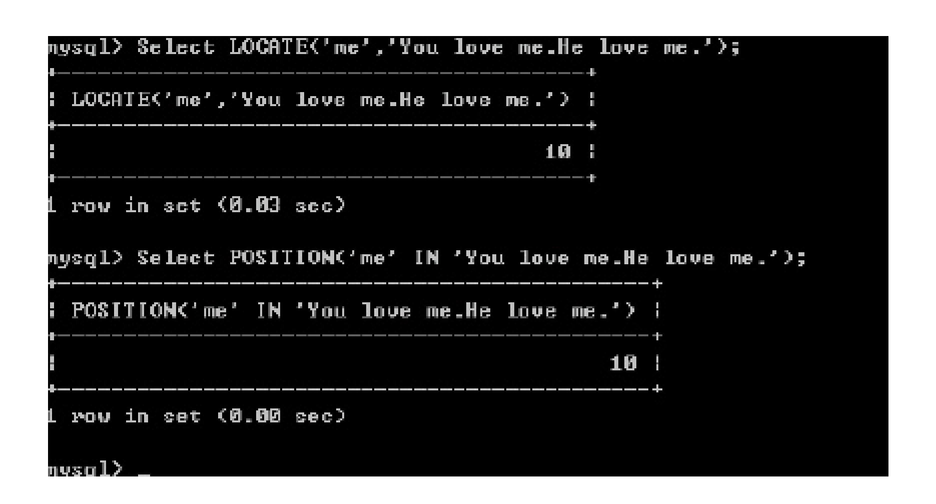
11.9.1　字符串函数的使用

例 11.31　在本章介绍了常用的字符串函数，当返回字符串‘me’在字符串‘You love me .He love me’中第一次出现的位置。对于字符串函数中的 LOCATE(s1,s)和 POSITION(s1 IN s)函数是从字符串 s 中获取 s1 的开始位置。**（实例位置：光盘\TM\Instance\11\11.31）**

代码如下：

```
Select LOCATE('me','You love me.He love me.');
Select POSITION('me' IN 'You love me.He love me.');
```

运行结果如图 11.31 所示。


```
mysql> Select LOCATE('me','You love me.He love me.');
+-------------------------------------+
| LOCATE('me','You love me.He love me.') |
+-------------------------------------+
|                                  10 |
+-------------------------------------+
1 row in set (0.03 sec)

mysql> Select POSITION('me' IN 'You love me.He love me.');
+---------------------------------------------+
| POSITION('me' IN 'You love me.He love me.') |
+---------------------------------------------+
|                                          10 |
+---------------------------------------------+
1 row in set (0.00 sec)

mysql> _
```

图 11.31　字符串函数的使用

11.9.2　查看当前数据库版本号

例 11.32　用 VERSION()函数返回数据库的版本号，connection_id()函数返回服务器的连接数，也就是到现在为止 MySQL 服务的连接次数；database()和 schema()函数返回当前数据库名。**（实例位置：光盘\TM\Instance\11\11.32）**

代码如下：

```
Select VERSION(),DATABASE(),USER(),CONNECTION_ID();
```

运行结果如图 11.32 所示。

11.9.3　生成 3 个 1～100 之间的随机整数

例 11.33　使用 ROUND(x)函数生成一个与数 x 最接近的整数，当然，也可以使用 FLOOR(x)函数来生成一个小于或者等于 x 的最大整数。RAND()函数产生的是随机数，所以每次执行的结果都会是不一样的。**（实例位置：光盘\TM\Instance\11\11.33）**

代码如下：

```
Select ROUND(RAND()*100),FLOOR(RAND()*100),CEILING(RAND()*100);
```

运行结果如图 11.33 所示。

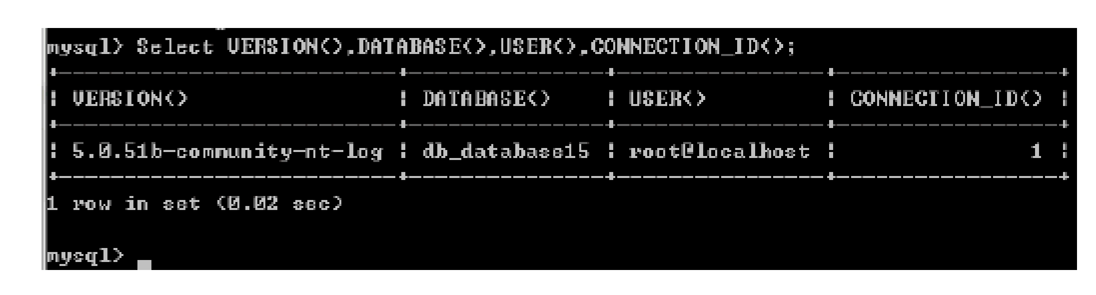


图 11.32　查看当前数据库版本号

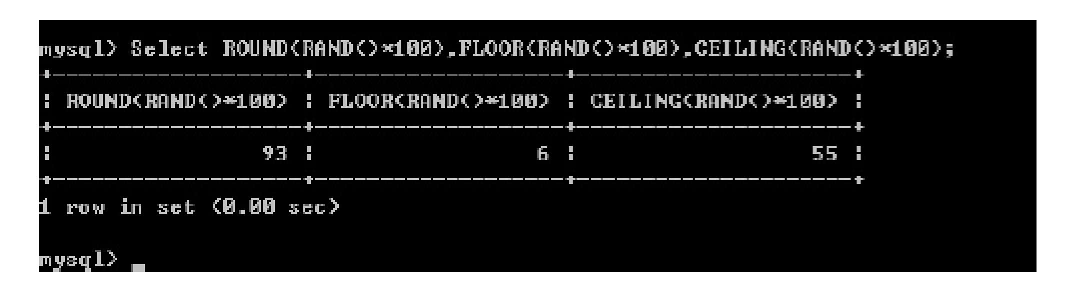


图 11.33　生成 3 个 1～100 之间的随机整数

11.9.4　数字函数的使用

例 11.34　将数字 1.98752895 保留到小数点后 4 位。**（实例位置：光盘\TM\Instance\11\11.34）**

代码如下：

```
select TRUNCATE(1.98752895,4);
```

运行结果如图 11.34 所示。

11.9.5　加密函数的使用

例 11.35　使用 MD5 对字符串‘mrsoft’进行加密的值。**（实例位置：光盘\TM\Instance\11\11.35）**

代码如下：

```
SELECT MD5('mrsoft');
```

运行结果如图 11.35 所示。

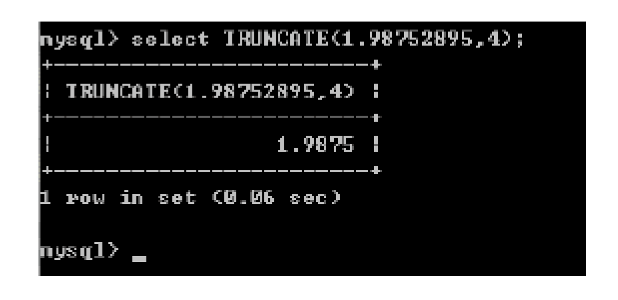


图 11.34　数学函数的使用

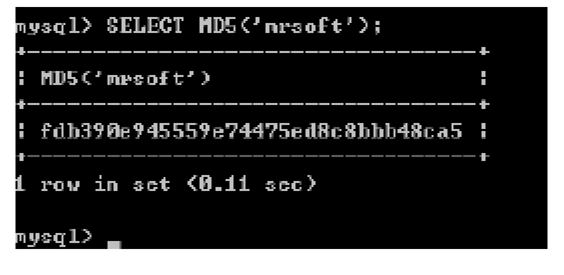


图 11.35　加密函数的使用

11.10　小　　结

本章介绍了 MySQL 数据库提供的内部函数。这些函数包括数学函数、字符串函数、日期和时间函数、

条件判断函数、系统信息函数和加密函数等。字符串和时间函数是本章重点介绍的内容。条件判断函数是本章的难点，因为条件判断函数涉及很多条件判断和跳转的语句。这些函数通常与 SELECT 语句一起使用，用来方便用户的查询。同时 INSERT、UPDATE、DELECT 语句和条件表达式也可以使用这些函数。读者一定要上机实际操作这些函数，这样可以对函数了解得更加透彻。

11.11　学习成果检验

1．计算 3 的 2 次方。**（实例位置：光盘\TM\Instance\11\11.36）**
2．将字符串‘MRBCCD’全部改为小写。**（实例位置：光盘\TM\Instance\11\11.37）**
3．将字符串‘mrbccd’逆序输出。**（实例位置：光盘\TM\Instance\11\11.38）**

第12章

MySQL 基本操作

（视频讲解：37 分钟）

MySQL 中提供了很多操作语句，通过这些语句可以对数据库、数据表，以及数据表中的数据等进行操作。本章将详细介绍 MySQL 的语句及其应用，包括创建、查看、选择、删除数据库；创建、修改、更名、删除数据表；插入、浏览、修改、删除记录和数据库的备份和恢复，这些是程序开发人员必须掌握的内容。

通过阅读本章内容，你可以：

- 掌握 MySQL 数据库操作
- 掌握 MySQL 数据表操作
- 掌握 MySQL 语句操作

12.1　MySQL 数据库操作

视频讲解：光盘\TM\Video\第 12 章\MySQL 数据库操作.exe

启动并连接 MySQL 服务器后，即可对 MySQL 数据库进行操作，操作 MySQL 数据库的方法非常简单。下面进行详细介绍。

12.1.1　创建数据库 CREATE DATABASE

使用 CREATE DATABASE 语句可以轻松创建 MySQL 数据库。语法如下：

```
CREATE  DATABASE  数据库名;
```

在创建数据库时，数据库命名有以下几项规则。

☑　不能与其他数据库重名，否则将发生错误。

☑　名称可以由任意字母、阿拉伯数字、下划线（_）和“$”组成，可以使用上述的任意字符开头，但不能使用单独的数字，否则会造成数据库与数值相混淆。

☑　名称最长可为 64 个字符，而别名最长可达 256 个字符。

☑　不能使用 MySQL 关键字作为数据库名、表名。

在默认情况下，Windows 下数据库名、表名的大小写是不敏感的，而在 Linux 下数据库名、表名的大小写是敏感的。为了便于数据库在平台间进行移植，建议读者采用小写来定义数据库名和表名。

例 12.1　通过 CREATE DATABASE 语句创建一个名称为 db_admin 的数据库，如图 12.1 所示。**（实例位置：光盘\TM\Instance\12\12.1）**

12.1.2　查看数据库 SHOW DATABASES

成功创建数据库后，可以使用 SHOW 命令来查看 MySQL 服务器中的所有数据库信息。语法如下：

```
SHOW  DATABASES;
```

例 12.2　在 12.1.1 节中创建了数据库 db_admin，下面使用 SHOW DATABASES 语句查看 MySQL 服务器中的所有数据库名称，如图 12.2 所示。**（实例位置：光盘\TM\Instance\12\12.2）**

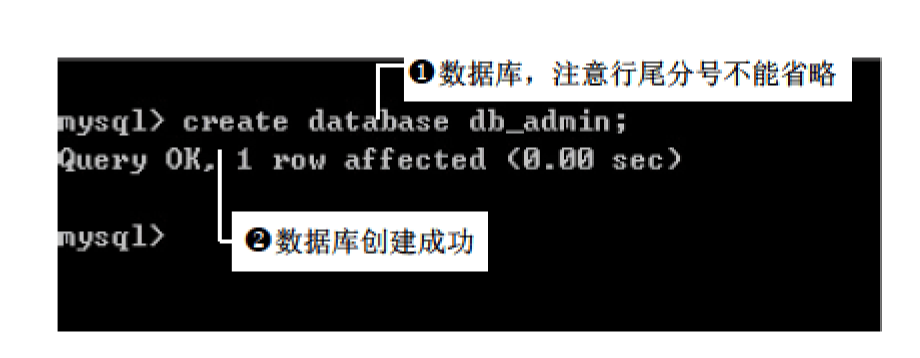


图 12.1　创建 MySQL 数据库

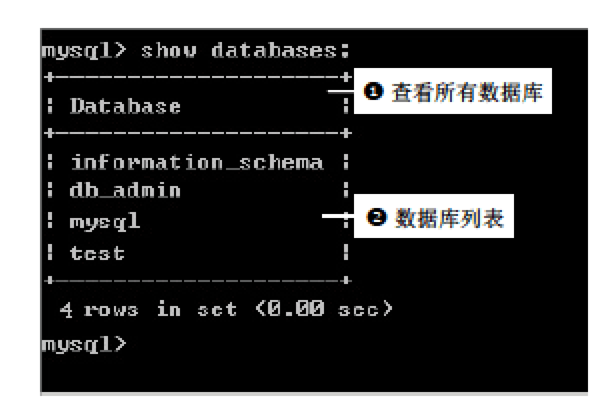


图 12.2　查看数据库

从图 12.2 运行的结果可以看出，MySQL 服务器中有 4 个数据库。

12.1.3　选择数据库 USE DATABASE

在上面的讲解中，虽然成功创建了数据库，但并不表示当前就在操作数据库 db_admin。可以使用 USE

语句选择一个数据库，使其成为当前默认数据库。语法如下：

```
USE  数据库名;
```

例 12.3　选择名称为 db_admin 的数据库，设置其为当前默认的数据库，如图 12.3 所示。（**实例位置：光盘\TM\Instance\12\12.3**）

12.1.4　删除数据库 DROP DATABASE

删除数据库的操作可以使用 DROP DATABASE 语句。语法如下：

```
DROP DATABASE  数据库名;
```

注意　删除数据库的操作应该谨慎使用，一旦执行该操作，数据库的所有结构和数据都会被删除，没有恢复的可能，除非数据库有备份。

例 12.4　通过 DROP DATABASE 语句删除名称为 db_admin 的数据库，如图 12.4 所示。（**实例位置：光盘\TM\Instance\12\12.4**）



图 12.3　选择数据库



图 12.4　删除数据库

12.2　MySQL 数据表操作

视频讲解：光盘\TM\Video\第 12 章\MySQL 数据表操作.exe

在对 MySQL 数据表进行操作前，必须首先使用 USE 语句选择数据库，才可在指定的数据库中对数据表进行操作，如创建数据表、修改表结构、更改数据表名或删除数据表等，否则是无法对数据表进行操作的。下面分别详细介绍对数据表的操作方法。

12.2.1　创建数据表 CREATE TABLE

创建数据表使用 CREATE TABLE 语句。语法如下：

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS]  数据表名
[(create_definition,…)][table_options] [select_statement]
```

CREATE TABLE 语句的参数说明如表 12.1 所示。

表 12.1　CREATE TABLE 语句的参数说明

关　键　字	说　　明
TEMPORARY	如果使用该关键字，表示创建一个临时表
IF NOT EXISTS	该关键字用于避免表存在时 MySQL 报告的错误
create_definition	表的列属性部分。MySQL 要求在创建表时，表要至少包含一列
table_options	表的一些特性参数
select_statement	SELECT 语句描述部分，用它可以快速地创建表

下面介绍列属性 create_definition 部分，每一列定义的具体格式如下：

```
col_name    type [NOT NULL | NULL] [DEFAULT default_value] [AUTO_INCREMENT]
            [PRIMARY KEY ] [reference_definition]
```

属性 create_definition 的参数说明如表 12.2 所示。

表 12.2　属性 create_definition 的参数说明

参　　数	说　　明
col_name	字段名
type	字段类型
NOT NULL | NULL	指出该列是否允许是空值，系统一般默认允许为空值，所以当不允许为空值时，必须使用 NOT NULL
DEFAULT default_value	表示默认值
AUTO_INCREMENT	表示是否是自动编号，每个表只能有一个 AUTO_INCREMENT 列，并且必须被索引
PRIMARY KEY	表示是否为主键。一个表只能有一个 PRIMARY KEY。如果表中没有 PRIMARY KEY，而某些应用程序需要 PRIMARY KEY，MySQL 将返回第一个没有任何 NULL 列的 UNIQUE 键，作为 PRIMARY KEY
reference_definition	为字段添加注释

以上是创建一个数据表的一些基础知识，看起来十分复杂，但在实际应用中使用最基本的格式创建数据表即可。具体格式如下：

```
create table table_name (列名 1  属性,列名 2  属性…);
```

例 12.5　使用 CREATE TABLE 语句在 MySQL 数据库 db_admin 中创建一个名为 tb_admin 的数据表，该表包括 id、user、password 和 createtime 等字段，如图 12.5 所示。**（实例位置：光盘\TM\Instance\12\12.5）**

12.2.2　查看表结构 SHOW COLUMNS 或 DESCRIBE

对于一个创建成功的数据表，可以使用 SHOW COLUMNS 或 DESCRIBE 语句查看指定数据表的表结构。下面分别对这两个语句进行介绍。

1. SHOW COLUMNS 语句

SHOW COLUMNS 语句的语法如下：

```
SHOW   [FULL] COLUMNS   FROM  数据表名  [FROM  数据库名];
```

或写成：

```
SHOW   [FULL] COLUMNS   FROM  数据表名.数据库名;
```

例 12.6　使用 SHOW COLUMNS 语句查看数据表 tb_admin 的结构，如图 12.6 所示。**（实例位置：光盘\TM\Instance\12\12.6）**

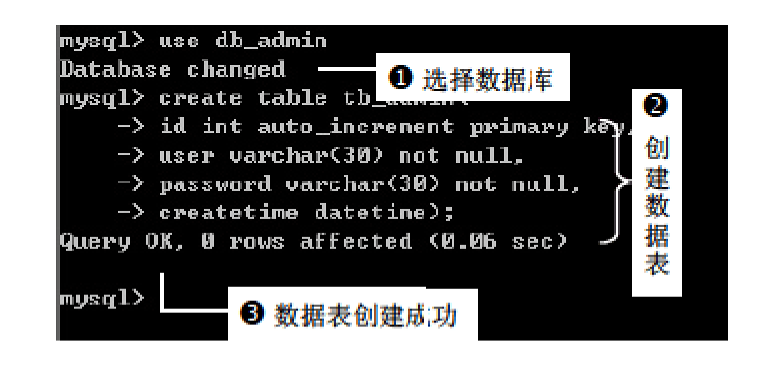


图 12.5　创建 MySQL 数据库

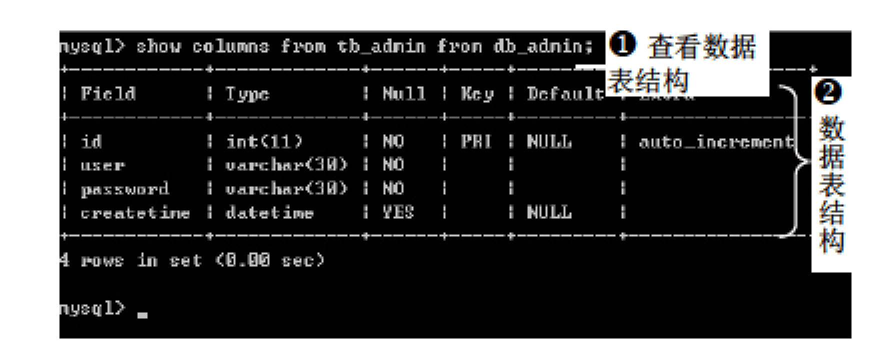


图 12.6　查看表结构

2. DESCRIBE 语句

DESCRIBE 语句的语法如下：

```
DESCRIBE 数据表名;
```

其中，DESCRIBE 可以简写成 DESC。在查看表结构时，也可以只列出某一列的信息。其语法格式如下：

```
DESCRIBE 数据表名 列名;
```

例 12.7 使用 DESCRIBE 语句的简写形式查看数据表 tb_admin 中的某一列信息，如图 12.7 所示。（**实例位置：光盘\TM\Instance\12\12.7**）

12.2.3 修改表结构 ALTER TABLE

修改表结构使用 ALTER TABLE 语句。修改表结构指增加或者删除字段、修改字段名称或者字段类型、设置或者取消主键/外键、设置或者取消索引以及修改表的注释等。语法如下：

```
Alter[IGNORE] TABLE 数据表名 alter_spec[,alter_spec]…
```



注意 当指定 IGNORE 时，如果出现重复关键的行，则只执行一行，其他重复的行被删除。

其中，alter_spec 子句定义要修改的内容，其语法如下：

```
alter_specification:
    ADD [COLUMN] create_definition [FIRST | AFTER column_name ]          //添加新字段
  | ADD INDEX [index_name] (index_col_name,...)                         //添加索引名称
  | ADD PRIMARY KEY (index_col_name,...)                                //添加主键名称
  | ADD UNIQUE [index_name] (index_col_name,...)                        //添加唯一索引
  | ALTER [COLUMN] col_name {SET DEFAULT literal | DROP DEFAULT}        //修改字段名称
  | CHANGE [COLUMN] old_col_name create_definition                      //修改字段类型
  | MODIFY [COLUMN] create_definition                                   //修改子句定义字段
  | DROP [COLUMN] col_name                                              //删除字段名称
  | DROP PRIMARY KEY                                                    //删除主键名称
  | DROP INDEX index_name                                               //删除索引名称
  | RENAME [AS] new_tbl_name                                            //更改表名
  | table_options
```

ALTER TABLE 语句允许指定多个动作，其动作间使用逗号分隔，每个动作表示对表的一个修改。

例 12.8 添加一个新的字段 email，类型为 varchar(50)，not null，将字段 user 的类型由 varchar(30)改为 varchar(40)。（**实例位置：光盘\TM\Instance\12\12.8**）

代码如下：

```
alter table tb_admin add email varchar(50) not null ,modify user varchar(40);
```

在命令模式下的运行情况如图 12.8 所示。



图 12.7 查看表的某一列信息

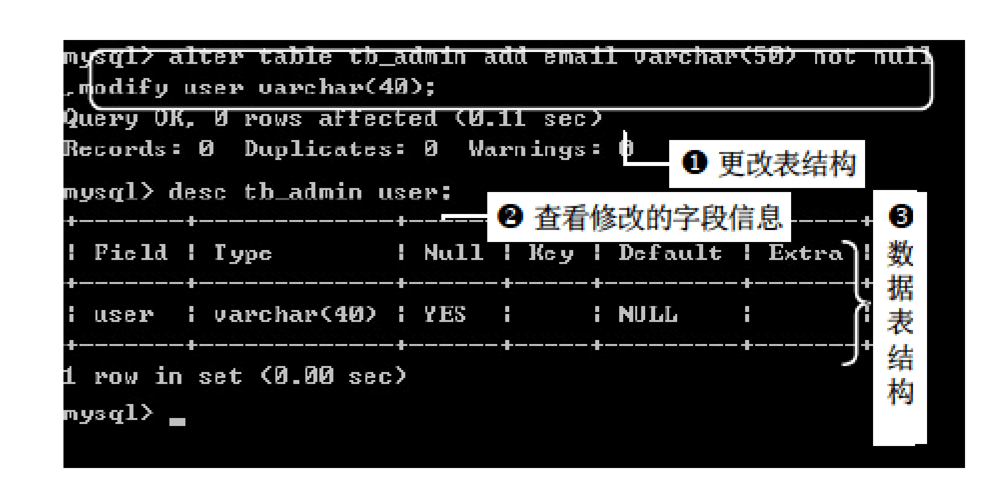


图 12.8 修改表结构

图 12.8 中只给出了修改 user 字段类型的结果，读者可以通过语句“mysql> show tb_admin;”查看整个表的结构，以确认 email 字段是否添加成功。


说明　通过 alter 修改表列，其前提是必须将表中数据全部删除，然后才可以修改表列。

12.2.4　重命名表 RENAME TABLE

重命名数据表使用 RENAME TABLE 语句。其语法如下：

```
RENAME TABLE 数据表名 1 To 数据表名 2
```


说明　该语句可以同时对多个数据表进行重命名，多个表之间以逗号“,”分隔。

例 12.9　重命名数据表 tb_admin，更名后的数据表为 tb_user，如图 12.9 所示。（**实例位置：光盘\TM\Instance\12\12.9**）

12.2.5　删除表 DROP TABLE

删除数据表的操作很简单，同删除数据库的操作类似，使用 DROP TABLE 语句即可实现。语法如下：

```
DROP TABLE 数据表名;
```

例 12.10　删除数据表 tb_user，如图 12.10 所示。（**实例位置：光盘\TM\Instance\12\12.10**）

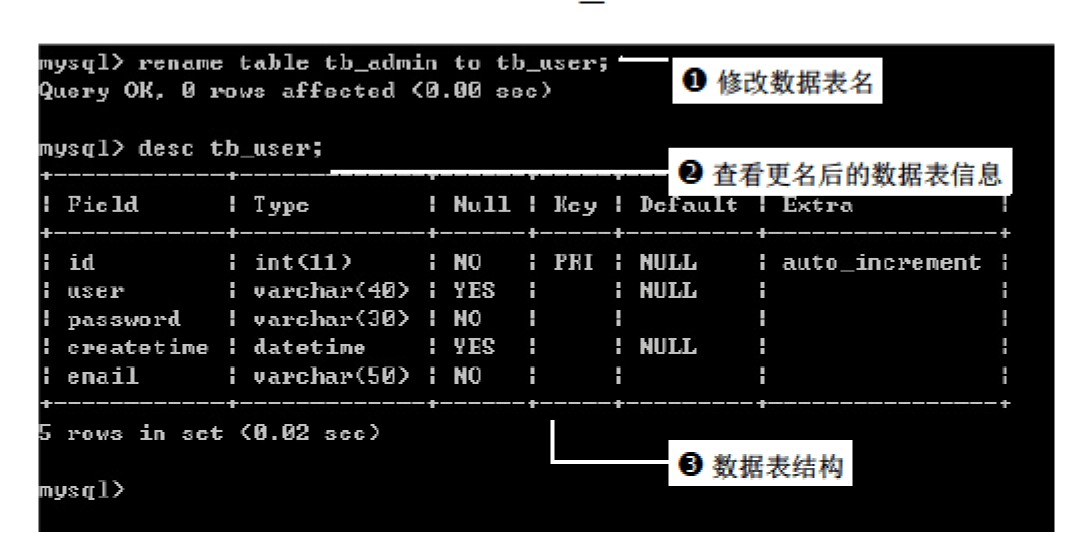



图 12.9　对数据表进行更名



图 12.10　删除数据表

注意　删除数据表的操作应该谨慎使用。一旦删除了数据表，表中的数据将会全部清除，没有备份则无法恢复。

在删除数据表的过程中，删除一个不存在的表将会产生错误，如果在删除语句中加入 IF EXISTS 关键字可避免此错误发生。格式如下：

```
DROP TABLE IF EXISTS 数据表名;
```

12.3　MySQL 语句操作

视频讲解：光盘\TM\Video\第 12 章\MySQL 语句操作.exe

在数据表中插入、浏览、修改和删除记录可以在 MySQL 命令行中使用 SQL 语句完成，下面介绍如何在 MySQL 命令行中执行基本的 SQL 语句。

12.3.1 插入记录 INSERT

在建立一个空的数据库和数据表时，首先需要考虑的是如何向数据表中添加数据，该操作可以使用 INSERT 语句来完成。语法如下：

```
insert  into 数据表名(column_name,column_name2, … ) values (value1, value2, … )
```

在 MySQL 中，一次可以同时插入多行记录，各行记录的值清单在 values 关键字后以逗号（,）分隔，而标准的 SQL 语句一次只能插入一行。

例 12.11 向管理员信息表 tb_admin 中插入一条数据信息，如图 12.11 所示。**（实例位置：光盘\TM\Instance\12\12.11）**



图 12.11 插入记录

12.3.2 查询数据库记录 SELECT

要从数据库中把数据查询出来，就要用到数据查询语句 SELECT。SELECT 语句是最常用的查询语句，其使用方式有些复杂，但功能很强大。其语法如下：

```
select [distinct] [concat(col1,":",col2) as col] selection_list     //要查询的内容，选择哪些列
from 数据表名                                                        //指定数据表
where primary_constraint                                             //查询时需要满足的条件，行必须满足的条件
group by grouping_columns                                            //如何对结果进行分组
order by sorting_cloumns                                             //如何对结果进行排序
having secondary_constraint                                          //查询时满足的第二条件
limit count                                                          //限定输出的查询结果
```

这就是 SELECT 查询语句的语法，下面对它的参数进行详细讲解。

1．selection_list

设置查询内容。如果要查询表中所有列，可以将其设置为“*”；如果要查询表中某一列或多列，则直接输入列名，并以“,”为分隔符。

例 12.12 查询 tb_mrbook 数据表中所有列和 user、pass 列。**（实例位置：光盘\TM\Instance\12\12.12）**

代码如下：

```
select * from tb_mrbook;                    //查询数据表中所有数据
select user,pass from tb_mrbook;            //查询数据表中 user 和 pass 列的数据
```

2．table_list（数据表名）

指定查询的数据表。既可以从一个数据表中查询，也可以从多个数据表中进行查询，多个数据表之间用“,”进行分隔，并且通过 WHERE 子句使用连接运算来确定表之间的联系。

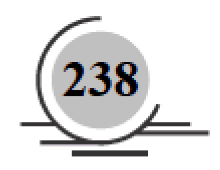
例 12.13 从 tb_mrbook 和 tb_bookinfo 数据表中查询“bookname=' PHP 开发实战宝典'”的作者和价格。**（实例位置：光盘\TM\Instance\12\12.13）**

其代码如下：

```
select tb_mrbook.id,tb_mrbook.bookname, author,price from tb_mrbook,tb_bookinfo
where tb_mrbook.bookname = tb_bookinfo.bookname and tb_bookinfo.bookname = 'PHP 开发实战宝典 ';
```

在上面的 SQL 语句中，因为两个表都有 id 字段和 bookname 字段，为了告诉服务器要显示的是哪个表中的字段信息，要加上前缀。语法如下：

```
表名.字段名
```



“tb_mrbook.bookname = tb_bookinfo.bookname”将表 tb_mrbook 和 tb_bookinfo 连接起来，叫做等同连接；如果不使用“tb_mrbook.bookname = tb_bookinfo.bookname”，那么产生的结果将是两个表的笛卡儿积，叫做全连接。

3. WHERE 条件语句

在使用查询语句时，如要从很多记录中查询出想要的记录，就需要一个查询条件。只有设定了查询条件，查询才有实际意义。设定查询条件应用的是 WHERE 子句。

WHERE 子句的功能非常强大，通过它可以实现很多复杂的条件查询。在使用 WHERE 子句时，需要使用一些比较运算符，常用的比较运算符如表 12.3 所示。

表 12.3　常用的 WHERE 子句比较运算符

运 算 符	名　称	示　例	运 算 符	名　称	示　例
=	等于	id=5	is not null	n/a	Id is not null
>	大于	id>5	between	n/a	Id between1 and 15
<	小于	id<5	in	n/a	Id in (3,4,5)
=>	大于等于	id=>5	not in	n/a	Name not in (shi,li)
<=	小于等于	id<=5	like	模式匹配	Name like ('shi%')
!=或<>	不等于	id!=5	not like	模式匹配	Name not like ('shi%')
is null	n/a	id is null	regexp	常规表达式	Name 正则表达式

表 12.3 中列举的是 WHERE 子句常用的比较运算符，示例中的 id 是记录的编号，name 是表中的用户名。

例 12.14　应用 WHERE 子句，查询 tb_mrbook 表，条件是 type（类别）为 PHP 的所有图书。**（实例位置：光盘\TM\Instance\12\12.14）**

代码如下：

```
select * from tb_mrbook where type = 'PHP';
```

4. GROUP BY 对结果分组

通过 GROUP BY 子句可以将数据划分到不同的组中，实现对记录进行分组查询。在查询时，所查询的列必须包含在分组的列中，目的是使查询到的数据没有矛盾。在与 AVG()或 SUM()函数一起使用时，GROUP BY 子句能发挥出最大作用。

例 12.15　查询 tb_mrbook 表，按照 type 进行分组，求每类图书的平均价格。**（实例位置：光盘\TM\Instance\12\12.15）**

代码如下：

```
select bookname,avg(price),type from tb_mrbook group by type;
```

5. DISTINCT 在结果中去除重复行

使用 DISTINCT 关键字，可以去除结果中重复的行。

例 12.16　查询 tb_mrbook 表，并在结果中去掉类型字段 type 中的重复数据。**（实例位置：光盘\TM\Instance\12\12.16）**

代码如下：

```
select distinct type from tb_mrbook;
```

6. ORDER BY 对结果排序

使用 ORDER BY 可以对查询的结果进行升序或降序排列。在默认情况下，ORDER BY 按升序输出结果。如果要按降序排列，可以使用 DESC 来实现。

在对含有 NULL 值的列进行排序时，如果是按升序排列，NULL 值将出现在最前面；如果是按降序排

列，NULL 值将出现在最后。

例 12.17　查询 tb_mrbook 表中的所有信息，按照 id 进行降序排列，并且只显示 3 条记录。**（实例位置：光盘\TM\Instance\12\12.17）**

其代码如下：

```
select * from tb_mrbook order by id desc limit 3;
```

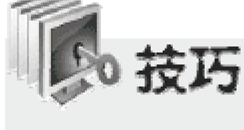
7．LIKE 模糊查询

LIKE 属于较常用的比较运算符，通过它可以实现模糊查询。它有两种通配符：“%”和下划线（_）。“%”可以匹配一个或多个字符，而“_”只匹配一个字符。

例 12.18　查找所有第二个字母是“h”的图书。**（实例位置：光盘\TM\Instance\12\12.18）**

代码如下：

```
select * from tb_mrbook where bookname like('_h%');
```



技巧　“p”和“入”都算作一个字符，这点上英文字母和中文没有什么区别。

8．CONCAT 联合多列

使用 CONCAT 函数可以联合多个字段，构成一个总的字符串。

例 12.19　把 tb_mrbook 表中的书名（bookname）和价格（price）合并到一起，构成一个新的字符串。**（实例位置：光盘\TM\Instance\12\12.19）**

代码如下：

```
select id,concat(bookname,":",price) as info,type from tb_mrbook;
```

其中合并后的字段名为 CONCAT 函数形成的表达式“concat(bookname,":",price)”，看上去十分复杂，通过 AS 关键字给合并字段取一个别名，这样看上去就很清晰。

9．LIMIT 限定结果行数

LIMIT 子句可以对查询结果的记录条数进行限定，控制它输出的行数。

例 12.20　查询 tb_mrbook 表，按照图书价格降序排列，显示 3 条记录。**（实例位置：光盘\TM\Instance\12\12.20）**

代码如下：

```
select * from tb_mrbook order by price desc limit 3;
```

使用 LIMIT 还可以从查询结果的中间部分取值。首先要定义两个参数，参数 1 是开始读取的第一条记录的编号（在查询结果中，第一个结果的记录编号是 0，而不是 1）；参数 2 是要查询记录的个数。

例 12.21　查询 tb_mrbook 表，从编号 1 开始（即从第 2 条记录）查询 4 个记录。**（实例位置：光盘\TM\Instance\12\12.21）**

代码如下：

```
select * from tb_mrbook where id limit 1,4;
```

10．使用函数和表达式

在 MySQL 中，还可以使用表达式来计算各列的值，作为输出结果。表达式还可以包含一些函数。

例 12.22　计算 tb_mrbook 表中各类图书的总价格。**（实例位置：光盘\TM\Instance\12\12.22）**

代码如下：

```
select sum(price) as total,type from tb_mrbook group by type;
```

在对 MySQL 数据库进行操作时，有时需要对数据库中的记录进行统计，如求平均值、最小值、最大值等，这时可以使用 MySQL 中的统计函数，常用统计函数如表 12.4 所示。



表 12.4　常用统计函数

名　　称	说　　明
Avg（字段名）	获取指定列的平均值
Count（字段名）	如指定了一个字段，则会统计出该字段中的非空记录。如在前面增加 DISTINCT，则会统计不同值的记录，相同的值当作一条记录。如使用 COUNT(*)则统计包含空值的所有记录数
Min（字段名）	获取指定字段的最小值
Max（字段名）	获取指定字段的最大值
Std（字段名）	指定字段的标准背离值
Stdtev（字段名）	与 Std 相同
Sum（字段名）	指定字段所有记录的总合

除了使用函数之外，还可以使用算术运算符、字符串运算符，以及逻辑运算符来构成表达式。

例 12.23　计算图书打八折之后的价格。（**实例位置：光盘\TM\Instance\12\12.23**）

代码如下：

```
select *, (price * 0.8) as '80%' from tb_mrbook;
```

12.3.3　修改记录 UPDATE

要执行修改的操作，可以使用 UPDATE 语句。其语法如下：

```
update  数据表名 set column_name = new_value1,column_name2 = new_value2, …where condition
```

其中，SET 子句指出要修改的列和它们给定的值；WHERE 子句是可选的，如果给出，将指定记录中哪行应该被更新，否则，所有的记录行都将被更新。

例 12.24　将管理员信息表 tb_admin 中用户名为 tsoft 的管理员密码 111 修改为 896552，如图 12.12 所示。（**实例位置：光盘\TM\Instance\12\12.24**）



图 12.12　修改指定条件的记录



更新时一定要保证 WHERE 子句的正确性，一旦 WHERE 子句出错，将会破坏所有改变的数据。

12.3.4　删除记录 DELETE

在数据库中，有些数据已经失去意义或者出现错误时就需要将它们删除，此时可以使用 DELETE 语句。其语法如下：

```
delete from  数据表名  where condition
```

注意：该语句在执行过程中，如果没有指定 where 条件，将删除所有的记录；如果指定了 where 条件，将按照指定的条件进行删除。

例 12.25 删除管理员数据表 tb_admin 中用户名为“小欣”的记录信息，如图 12.13 所示。（**实例位置：光盘\TM\Instance\12\12.25**）



图 12.13 删除数据表中指定的记录

注意：在实际应用中，执行删除操作时，执行删除的条件一般应该为数据的 id，而不是具体某个字段值，这样可以避免一些不必要的错误发生。

12.4 实　　战

视频讲解：光盘\TM\Video\第 12 章\视频 12.4\实战.exe

12.4.1 操作 teacher 表

例 12.26 在数据库中创建一个 teacher 表，并且将 teacher 表的 name 字段的数据类型改为 varchar(30)，将 birthday 字段的位置改到 sex 字段的前面。（**实例位置：光盘\TM\Instance\12\12.26**）

代码如下：

```
Create table teacher(id int(4) not null primary key auto_increment,
num int(10) not null ,
name varchar(20) not null,
sex varchar(4) not null,
birthday datetime,
address varchar(50)
);
Alter table teacher modify name varchar(30) not null;
Alter table teacher modify birthday datetime after name;
```

运行效果如图 12.14 所示。

12.4.2 登录数据库系统

例 12.27 本实例在命令行中登录 MySQL 数据库管理系统，输入用户名、连接的主机、密码，进入到 mysql 命令行中。（**实例位置：光盘\TM\Instance\12\12.27**）

代码如下：

```
mysql –uroot –h127.0.0.1 –p111
```

运行结果如图 12.15 所示。



```
mysql> Create table teacher(id int(4) not null primary key auto_increment,
    -> num int(10) not null ,
    -> name varchar(20) not null,
    -> sex varchar(4) not null,
    -> birthday datetime,
    -> address varchar(50)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> alter table teacher modify name varchar(30) not null;
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table teacher modify birthday datetime after name;
Query OK, 0 rows affected (0.16 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> _
```

图 12.14　操作 teacher 表

```
C:\Users\Administrator>mysql -uroot -h127.0.0.1 -p111
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 24
Server version: 5.5.12 MySQL Community Server (GPL)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> _
```

图 12.15　登录数据库系统

12.4.3　读取 MySQL 数据库中数据（PHP 语言）

例 12.28　读取数据库中的数据通过 mysql_query()函数和 select 语句查询数据，使用 mysql_fetch_assoc()函数将查询结果返回到数组中。本例读取 tb_news 表中的新闻信息。（**实例位置：光盘\TM\Instance\12\12.28**）

代码如下：

```
<?php
/*连接数据库*/
$conn=mysql_connect("localhost","root","111");                //连接数据库服务器
mysql_select_db("db_database12",$conn);                       //选择数据库
mysql_query("set names utf8");                                //设置编码格式
$arr=mysql_query("select * from tb_news",$conn);              //执行查询语句
/*使用 while 语句循环 mysql_fetch_assoc()函数返回的数组*/
while($result=mysql_fetch_assoc($arr)){                       //循环输出查询结果
?>
     <tr>
     <td height="25"><?php echo $result['name'];?>  </td>     //输出新闻标题
          <td height="25"><?php echo $result['news'];?> </td>      //输出新闻内容
    </tr>
<?php
  }                                                                //结束 while 循环
?>
```

运行效果如图 12.16 所示。

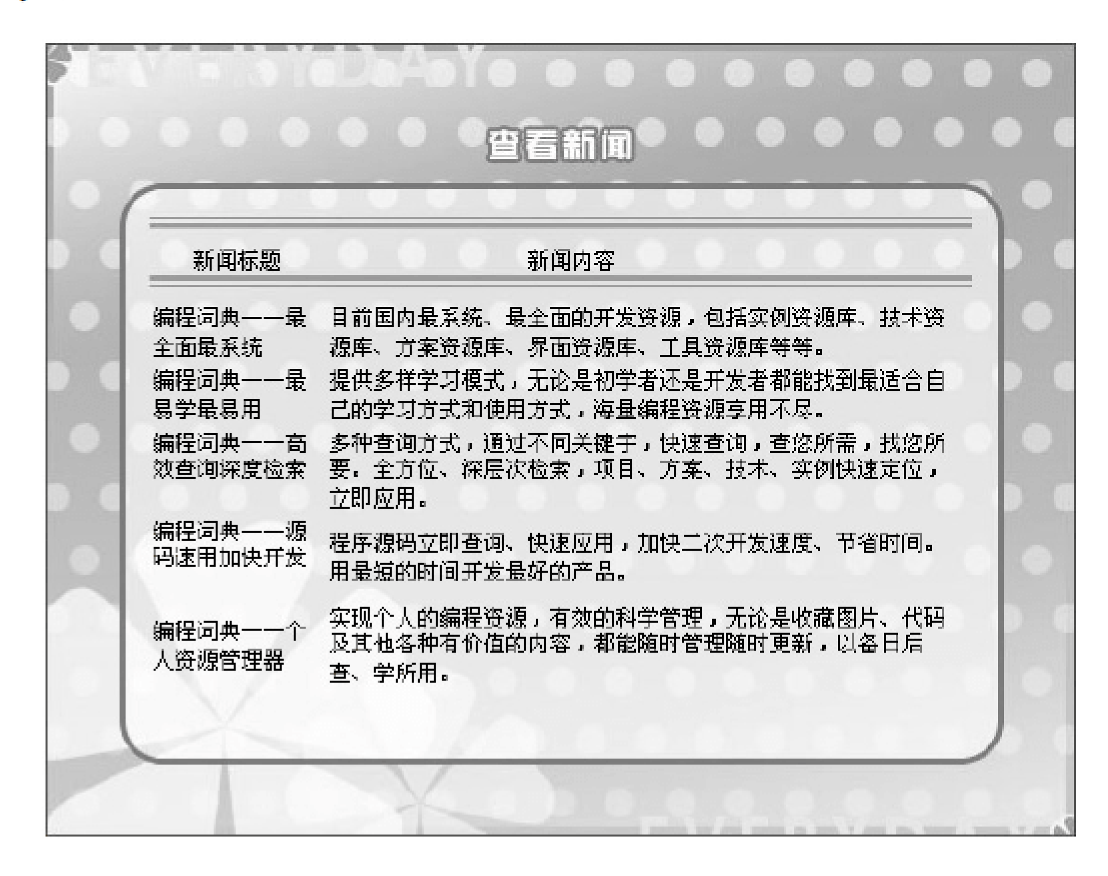


图 12.16　浏览新闻信息

12.4.4 备份和恢复 MySQL 数据库（Java 语言）

例 12.29 本实例使用 Mysqldump 命令备份数据库，运行结果如图 12.17 所示。对于大型 Web 项目来说，具有良好的数据备份功能是非常重要的，数据恢复可以在数据遭到破坏后，将备份的数据恢复到数据库系统中，以此来保证系统的正常运行，以免数据丢失带来损失。数据库恢复的运行结果如图 12.18 所示。**（实例位置：光盘\TM\Instance\12\12.29）**

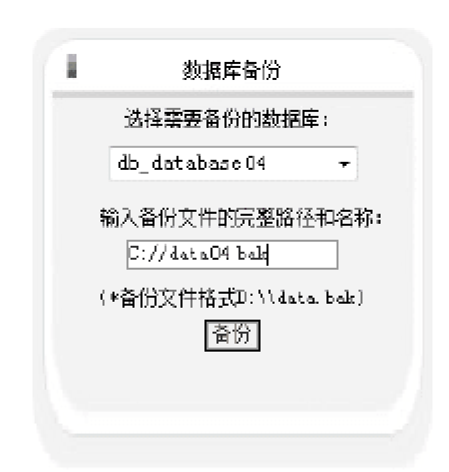


图 12.17 数据库备份的运行结果



图 12.18 数据库恢复的运行结果

数据库备份的实现过程如下。

（1）创建 UserDao 类，定义数据库连接、更新和关闭的方法，并且在该类中定义数据库备份的方法 mysqldump()。本实例中连接的是 MySQL 的系统数据库 information_schema。

（2）创建 index.jsp 页面，读取 information_schema 数据库 SCHEMATA 数据表中的数据；创建 form 表单，添加下拉列表，将 SCHEMATA 数据表中的数据添加到下拉列表中；添加文本框，设置备份文件的存储位置和路径。其关键代码如下：

```
<%@ page contentType="text/html; charset=gbk" language="java" import="java.sql.*" errorPage="" %>
    <jsp:useBean id="dao" scope="request" class="com.pkh.dao.UserDao"/>
<%
    ResultSet rs = dao.selectStatic("SELECT schema_name FROM SCHEMATA");  //执行查询操作，获取结果集
    String path = request.getParameter("path");                            //获取备份文件存储的位置
    String dbase = request.getParameter("select");                         //获取指定备份的数据库
    if("备份".equals(request.getParameter("Submit"))){
        if (path != null && dbase != null) {
            if (dao.mysqldump(dbase,path)) {                      //调用方法，执行数据库的备份
                out.print("<script language='javascript'>alert('备份完成');</script>");
            }
        }
    }
%>
<form name="form1" method="post" action="">
    <select name="select">
    <%
        while (rs.next()) {
            String database = rs.getString(1);
            out.println("<option id='" + database + "'>" + database + "</option>");
        }
        dao.closeConnection();
    %>
    </select>
    <input type="text" name="path">
    <input type="submit" name="Submit" value="备份">
</form>
```

数据库恢复的关键代码如下：

```
<%@ page contentType="text/html; charset=gbk" language="java" import="java.sql.*" errorPage="" %>
    <jsp:useBean id="dao" scope="request" class="com.pkh.dao.UserDao"/>
<%
    ResultSet rs = dao.selectStatic("SELECT schema_name FROM SCHEMATA");      //执行查询语句
    String path = request.getParameter("path");                              //获取恢复文件存储路径
    String dbase = request.getParameter("select");                           //获取要恢复的数据库
    if ("恢复".equals(request.getParameter("Submit"))) {
        if (path != null && dbase != null) {
            if (dao.mysqlresume(dbase,path)) {                               //执行恢复操作
                out.print("<script language='javascript'>alert('恢复完成');</script>");
            }
        }
    }
%>
<form name="form1" method="post" action="">
    <input name="path" type="file" size="18">
    <select name="select">
    <%
        while (rs.next()) {
            String database = rs.getString(1);
            out.println("<option id='" + database + "'>" + database + "</option>");
        }
        dao.closeConnection();
    %>
    </select>
    <input type="submit" name="Submit" value="恢复">
</form>
```

12.4.5　查看表详细结构语句 SHOW CREATE TABLE

例 12.30　下面用 SHOW CREATE TABLE 语句查看 TEACHER 表的定义。**（实例位置：光盘\TM\Instance\12\12.30）**

代码如下：

```
SHOW CREATE TABLE TEACHER \G
```

运行结果如图 12.19 所示。



图 12.19　查看表详细结构

12.5　小　　结

本章介绍了创建表和库、查看表结构、修改表及删除表和库的方法。创建和修改表和库的内容比较重

要，这两部分需要不断的练习。只有通过实践练习，才会对这两部分了解得更加透彻。而且，这两部分很容易出现语法错误，必须在练习中掌握正确的语法规则。创建表和修改表后一定要查看表的结构，这样可以确认操作是否正确。删除表时一定要特别小心，因为删除表的同时会删除表中的所有记录。

12.6 学习成果检验

1. 如何查看数据库？**（实例位置：光盘\TM\Instance\12\12.31）**
2. 如何创建数据库？**（实例位置：光盘\TM\Instance\12\12.32）**
3. 如何把原来的数据表重命名？**（实例位置：光盘\TM\Instance\12\12.33）**
4. 如何删除数据表？**（实例位置：光盘\TM\Instance\12\12.34）**

第13章

MySQL 数据查询

（视频讲解：60 分钟）

数据查询是指从数据库中获取所需要的数据。数据查询是数据库操作中最常用，也是最重要的操作。用户可以根据自己对数据的需求，使用不同的查询方式。通过不同的查询方式，可以获得不同的数据。在 MySQL 中是使用 SELECT 语句来查询数据的。本章将对查询语句的基本语法、在单表上查询数据、使用聚合函数查询数据、合并查询结果等内容进行详细地讲解，使读者轻松了解查询数据的语句。

通过阅读本章内容，你可以：

- 了解 MySQL 的单表查询
- 了解使用聚合函数实现数据查询
- 掌握连接查询和子查询
- 掌握合并查询的使用
- 掌握为表和字段取别名的用法
- 掌握正则表达式的使用方法

13.1 基本查询语句

视频讲解：光盘\TM\Video\第 13 章\基本查询语句.exe

SELECT 语句是最常用的查询语句，它的使用方式有些复杂，但功能是相当强大的。SELECT 语句的基本语法如下：

```
select selection_list                    //要查询的内容，选择哪些列
from  数据表名                           //指定数据表
where primary_constraint                 //查询时需要满足的条件，行必须满足的条件
group by grouping_columns                //如何对结果进行分组
order by sorting_cloumns                 //如何对结果进行排序
having secondary_constraint              //查询时满足的第二条件
limit count                              //限定输出的查询结果
```

其中使用的子句将在后面逐个介绍。下面先介绍 SELECT 语句的简单应用。

（1）使用 SELECT 语句查询一个数据表

使用 SELECT 语句时，首先要确定所要查询的列。“*”代表所有的列，例如，查询 db_database13 数据库 user 表中的所有数据。代码如下：

```
mysql> use db_database13
Database changed
mysql> select * from user;
+----+------+----------+--------------+
| id | user | lxdh     | jtdz         |
+----+------+----------+--------------+
|  1 | mr   | 12345678 | 长春市 |
|  2 | mrsoft | 87654321 | 四平市     |
+----+------+----------+--------------+
2 rows in set (0.00 sec)
```

这是查询整个表中所有列的操作，还可以针对表中的某一列或多列进行查询。

（2）查询表中的一列或多列

针对表中的多列进行查询，只要在 select 后面指定要查询的列名即可，多列之间用“,”分隔。例如，查询 user 表中的 id 和 lxdh。代码如下：

```
mysql> select id , lxdh from user ;
+----+----------+
| id | lxdh     |
+----+----------+
|  1 | 12345678 |
|  2 | 87654321 |
+----+----------+
2 rows in set (0.00 sec)
```

13.2 单 表 查 询

视频讲解：光盘\TM\Video\第 13 章\单表查询.exe

单表查询是指从一张表中查询所需要的数据。所有查询操作都比较简单。



13.2.1　查询所有字段

查询所有字段是指查询表中所有字段的数据。这种方式可以将表中所有字段的数据都查询出来。在 MySQL 中可以使用“*”代表所有的列，即可查出所有的字段。语法格式如下：

```
SELECT * FROM 表名;
```

其应用已经在 13.1 基本查询语句中领教过，这里不再赘述了。

13.2.2　查询指定字段

查询指定字段可以使用下面的语法格式：

```
SELECT 字段名 FROM 表名;
```

如果是查询多个字段，可以使用“,”对字段进行分隔。

例 13.1　查询 db_database13 数据库 tb_login 表中 “user”和“pwd”两个字段。**（实例位置：光盘\TM\Instance\13\13.1）**

SELECT 查询语句如下：

```
SELECT user,pwd FROM tb_login;
```

查询结果如图 13.1 所示。



```
mysql> select user,pwd from tb_login;
+------+--------+
| user | pwd    |
+------+--------+
| mrkj | mrkj   |
| lx   | lx     |
| mr   | mrsoft |
+------+--------+
3 rows in set (0.00 sec)

mysql>
```

图 13.1　查询指定字段的数据

13.2.3　查询指定数据

如果要从很多记录中查询出指定的记录，那么就需要一个查询的条件。设定查询条件应用的是 WHERE 子句。通过它可以实现很多复杂的条件查询。在使用 WHERE 子句时，需要使用一些比较运算符来确定查询的条件。常用比较运算符如表 13.1 所示。

表 13.1　比较运算符

运 算 符	名　称	示　例	运 算 符	名　称	示　例
=	等于	Id=5	Is not null	n/a	Id is not null
>	大于	Id>5	Between	n/a	Id between1 and 15
<	小于	Id<5	In	n/a	Id in (3,4,5)
=>	大于等于	Id=>5	Not in	n/a	Name not in (shi,li)
<=	小于等于	Id<=5	Like	模式匹配	Name like (‘shi%’)
!=或<>	不等于	Id!=5	Not like	模式匹配	Name not like (‘shi%’)
Is null	n/a	Id is null	Regexp	常规表达式	Name 正则表达式

表 13.1 中列举的是 WHERE 子句常用的比较运算符，示例中的 id 是记录的编号，name 是表中的用户名。

例 13.2　应用 where 子句，查询 tb_login 表，条件是 user（用户名）为 mr。**（实例位置：光盘\TM\Instance\13\13.2）**

```
mysql> select * from tb_login where user='mr';
+----+------+--------+---------+----------+
| id | user | pwd    | section | name     |
+----+------+--------+---------+----------+
|  8 | mr   | mrsoft | PHP     | 明日科技 |
+----+------+--------+---------+----------+
1 row in set (0.00 sec)

mysql>
```

图 13.2　查询指定数据

代码如下：

```
select * from tb_login where user = 'mr';
```

查询结果如图 13.2 所示。

13.2.4　带 IN 关键字的查询

IN 关键字可以判断某个字段的值是否在指定的集合中。如果字段的值在集合中，则满足查询条件，该记录将被查询出来；如果不在集合中，则不满足查询条件。其语法格式如下：

```
SELECT * FROM 表名 WHERE 条件 [NOT] IN(元素 1,元素 2,…,元素 n);
```

☑　NOT：可选参数，加上 NOT 表示不在集合内满足条件。

☑　元素：集合中的元素，各元素之间用逗号隔开，字符型元素需要加上单引号。

例 13.3　应用 IN 关键字查询 tb_login 表中 user 字段为 mr 和 lx 的记录。（**实例位置：光盘\TM\Instance\13\13.3**）

查询语句如下：

```
SELECT * FROM tb_login WHERE user IN('mr','lx');
```

查询结果如图 13.3 所示。

例 13.4　使用 NOT IN 关键字查询 tb_login 表中 user 字段不为 mr 和 lx 的记录。（**实例位置：光盘\TM\Instance\13\13.4**）

查询语句如下：

```
SELECT * FROM tb_login WHERE user NOT IN('mr','lx');
```

查询结果如图 13.4 所示。

```
mysql> select * from tb_login where user IN('mr','lx');
+----+------+--------+-------------+--------+
| id | user | pwd    | section     | name   |
+----+------+--------+-------------+--------+
|  6 | lx   | lx     | PHP程序开发部 | 明日科技 |
|  8 | mr   | mrsoft | PHP         | 明日科技 |
+----+------+--------+-------------+--------+
2 rows in set (0.00 sec)

mysql>
```

图 13.3　使用 IN 关键字查询



图 13.4　使用 NOT IN 关键查询

13.2.5　带 BETWEEN AND 的范围查询

BETWEEN AND 关键字可以判断某个字段的值是否在指定的范围内。如果字段的值在指定范围内，则满足查询条件，该记录将被查询出来；如果不在指定范围内，则不满足查询条件。其语法如下：

```
SELECT * FROM 表名 WHERE 条件 [NOT] BETWEEN 取值 1 AND 取值 2;
```

☑　NOT：可选参数，加上 NOT 表示不在指定范围内满足条件。

☑　取值 1：范围的起始值。

☑　取值 2：范围的终止值。

例 13.5　查询 tb_login 表中 id 值在 5～7 之间的数据。（**实例位置：光盘\TM\Instance\13\13.5**）

查询语句如下：

```
SELECT * FROM tb_login WHERE id BETWEEN 5 AND 7;
```

查询结果如图 13.5 所示。

如果要查询 tb_login 表中 id 值不在 5～7 之间的数据，则可以通过 NOT BETWEEN AND 来完成。其查询语句如下：

```
SELECT * FROM tb_login WHERE id NOT BETWEEN 5 AND 7;
```

13.2.6　带 LIKE 的字符匹配查询

LIKE 属于较常用的比较运算符，通过它可以实现模糊查询。它有两种通配符：“%”和下划线“_”：“%”可以匹配一个或多个字符，可以代表任意长度的字符串，长度可以为 0。例如，“明%技”表示

以“明”开头，以“技”结尾的任意长度的字符串。该字符串可以代表明日科技、明日编程科技、明日图书科技等字符串。

“_”只匹配一个字符。例如，m_n 表示以 m 开头，以 n 结尾的 3 个字符。中间的“_”可以代表任意一个字符。

例 13.6　查询 tb_login 表中 user 字段中包含 mr 字符串的数据。（**实例位置：光盘\TM\Instance\13\13.6**）

查询语句如下：

```
select * from tb_login where user like '%mr%';
```

查询结果如图 13.6 所示。

```
mysql> select * from tb_login where id BETWEEN 5 and 7;
+----+------+------+---------------+----------+
| id | user | pwd  | section       | name     |
+----+------+------+---------------+----------+
|  6 | lx   | lx   | PHP程序开发部 | 明日科技 |
|  7 | mrkj | mrkj | 程序开发      | 明日科技 |
+----+------+------+---------------+----------+
2 rows in set (0.00 sec)

mysql>
```

图 13.5　使用 BETWEEN AND 关键字查询



```
mysql> select * from tb_login where user like '%mr%';
+----+------+--------+----------+----------+
| id | user | pwd    | section  | name     |
+----+------+--------+----------+----------+
|  7 | mrkj | mrkj   | 程序开发 | 明日科技 |
|  8 | mr   | mrsoft | PHP      | 明日科技 |
+----+------+--------+----------+----------+
2 rows in set (0.00 sec)
```

图 13.6　模糊查询

13.2.7　用 IS NULL 关键字查询空值

IS NULL 关键字可以用来判断字段的值是否为空值（NULL）。如果字段的值是空值，则满足查询条件，该记录将被查询出来；如果字段的值不是空值，则不满足查询条件。其语法格式如下：

```
IS [NOT] NULL
```

其中，NOT 是可选参数，加上 NOT 表示字段不是空值时满足条件。

例 13.7　下面使用 IS NULL 关键字查询 db_database13 数据库的 tb_book 表中 row 字段的值为空的记录。（**实例位置：光盘\TM\Instance\13\13.7**）

查询语句如下：

```
SELECT books,row FROM tb_book WHERE row IS NULL;
```

查询结果如图 13.7 所示。

13.2.8　带 AND 的多条件查询

AND 关键字可以用来联合多个条件进行查询。使用 AND 关键字时，只有同时满足所有查询条件的记录会被查询出来。如果不满足这些查询条件的其中一个，这样的记录将被排除掉。AND 关键字的语法格式如下：

```
select * from  数据表名  where  条件 1 and  条件 2 [...AND  条件表达式 n];
```

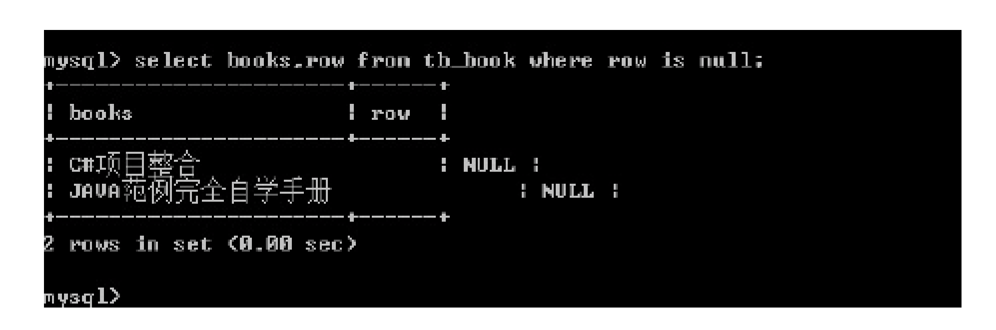
AND 关键字连接两个条件表达式，可以同时使用多个 AND 关键字来连接多个条件表达式。

例 13.8　下面查询 tb_login 表中 user 字段值为 mr，并且 section 字段值为 PHP 的记录。（**实例位置：光盘\TM\Instance\13\13.8**）

查询语句如下：

```
select * from tb_login where user='mr' and section='php';
```

查询结果如图 13.8 所示。



```
mysql> select books,row from tb_book where row is null;
+----------------------+------+
| books                | row  |
+----------------------+------+
| C#项目整合           | NULL |
| JAVA范例完全自学手册 | NULL |
+----------------------+------+
2 rows in set (0.00 sec)

mysql>
```

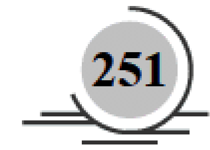
图 13.7　查询 tb_book 表中 row 字段值为空的记录



```
mysql> select * from tb_login where user='mr' and section='php';
+----+------+--------+---------+----------+
| id | user | pwd    | section | name     |
+----+------+--------+---------+----------+
|  8 | mr   | mrsoft | PHP     | 明日科技 |
+----+------+--------+---------+----------+
1 row in set (0.01 sec)

mysql>
```

图 13.8　使用 AND 关键字实现多条件查询



13.2.9　带 OR 的多条件查询

OR 关键字也可以用来联合多个条件进行查询，但是与 AND 关键字不同，OR 关键字只要满足查询条件中的一个，那么此记录就会被查询出来；如果不满足这些查询条件中的任何一个，这样的记录将被排除掉。OR 关键字的语法格式如下：

```
select * from 数据表名 where 条件 1 OR 条件 2 [...OR 条件表达式 n];
```

OR 可以用来连接两个条件表达式。而且，可以同时使用多个 OR 关键字连接多个条件表达式。

例 13.9　下面查询 tb_login 表中 section 字段的值为 PHP 或者“程序开发”的记录。（**实例位置：光盘\TM\Instance\13\13.9**）

查询语句如下：

```
select * from tb_login where section='php' or section='程序开发';
```

查询结果如图 13.9 所示。



图 13.9　使用 OR 关键字实现多条件查询

13.2.10　用 DISTINCT 关键字去除结果中的重复行

使用 DISTINCT 关键字可以去除查询结果中的重复记录。语法格式如下：

```
select distinct 字段名 from 表名;
```

例 13.10　下面使用 distinct 关键字去除 tb_login 表中 name 字段中的重复记录。（**实例位置：光盘\TM\Instance\13\13.10**）

查询语句如下：

```
select distinct name from tb_login;
```

查询结果如图 13.10 所示。去除重复记录前的 name 字段值如图 13.11 所示。

```
mysql> select distinct name from tb_login;
+----------+
| name     |
+----------+
| 高经理   |
| 明日科技 |
+----------+
2 rows in set (0.00 sec)

mysql>
```

图 13.10　使用 DISTINCT 关键字去除结果中的重复行

```
mysql> select name from tb_login;
+----------+
| name     |
+----------+
| 高经理   |
| 明日科技 |
| 明日科技 |
+----------+
3 rows in set (0.00 sec)

mysql> _
```

图 13.11　去除重复记录前的 name 字段值

13.2.11　用 ORDER BY 关键字对查询结果排序

使用 ORDER BY 关键字可以对查询的结果进行升序（ASC）和降序（DESC）排列，在默认情况下，ORDER BY 按升序输出结果。如果要按降序排列可以使用 DESC 来实现。语法格式如下：

```
ORDER BY 字段名 [ASC|DESC];
```

☑　ASC：表示按升序进行排序。

☑　DESC：表示按降序进行排序。



说明 如果对含有 NULL 值的列进行排序时，如果是按升序排列，NULL 值将出现在最前面；如果是按降序排列，NULL 值将出现在最后。

例 13.11　查询 tb_login 表中的所有信息，按照 id 进行降序排列。**（实例位置：光盘\TM\Instance\13\13.11）**

查询语句如下：

```
select * from tb_login order by id desc;
```

查询结果如图 13.12 所示。



```
mysql> select * from tb_login order by id desc;
+----+------+--------+------------------+-----------+
| id | user | pwd    | section          | name      |
+----+------+--------+------------------+-----------+
|  8 | mr   | mrsoft | PHP              | 明日科技 |
|  7 | mrkj | mrkj   | 程序开发 |           |
|  6 | lx   | lx     | PHP程序开发部 | 明日科技 |
+----+------+--------+------------------+-----------+
3 rows in set (0.03 sec)
```

图 13.12　按 id 编号进行降序排列

13.2.12　用 GROUP BY 关键字分组查询

通过 GROUP BY 子句可以将数据划分到不同的组中，实现对记录进行分组查询。在查询时，所查询的列必须包含在分组的列中，目的是使查询到的数据没有矛盾。

1．使用 GROUP BY 关键字来分组

单独使用 GROUP BY 关键字，查询结果只显示每组的一条记录。

例 13.12　使用 GROUP BY 关键字对 tb_book 表中 talk 字段进行分组查询。**（实例位置：光盘\TM\Instance\13\13.12）**

查询语句如下：

```
select id,books,talk from tb_book GROUP BY talk;
```

查询结果如图 13.13 所示。

为了使分组更加直观明了，下面查询 tb_book 表中的记录，查询结果如图 13.14 所示。

```
mysql> select id,books,talk from tb_book GROUP BY talk;
+----+----------------+------+
| id | books          | talk |
+----+----------------+------+
| 27 | PHP典型模块 | .net |
| 26 | JAVA典型模块 | JAVA |
| 30 | aa             | PHP  |
| 29 | aaaa           | 其它 |
+----+----------------+------+
4 rows in set (0.00 sec)

mysql>
```

图 13.13　使用 GROUP BY 关键字进行分组查询



```
mysql> select id,books,talk,user from tb_book;
+----+----------------------+------+------+
| id | books                | talk | user |
+----+----------------------+------+------+
| 26 | JAVA典型模块 | JAVA | mr   |
| 27 | PHP典型模块 | .net | mr   |
| 28 | C#项目整合 | .net | mr   |
| 29 | aaaa                 | 其它 | lx   |
| 30 | aa                   | PHP  | lx   |
| 25 | JAVA范例完全自学手册 | JAVA | mr   |
+----+----------------------+------+------+
6 rows in set (0.02 sec)

mysql>
```

图 13.14　tb_book 表中的记录

2．GROUP BY 关键字与 GROUP_CONCAT()函数一起使用

使用 GROUP BY 关键字和 GROUP_CONCAT()函数查询，可以将每个组中的所有字段值都显示出来。

例 13.13　下面使用 GROUP BY 关键字和 GROUP_CONCAT()函数对 tb_book 表中的 talk 字段进行分组查询。**（实例位置：光盘\TM\Instance\13\13.13）**

查询语句如下：

```
select id,books,GROUP_CONCAT(talk) from tb_book GROUP BY talk;
```

查询结果如图 13.15 所示。

3．按多个字段进行分组

使用 GROUP BY 关键字也可以按多个字段进行分组。

例 13.14 下面对 tb_book 表中的 user 和 sort 字段进行分组，分组过程中，按照 user 进行分组。**（实例位置：光盘\TM\Instance\13\13.14）**

查询语句如下：

```
select id,books ,user from tb_book GROUP BY user;
```

查询结果如图 13.16 所示。

mysql> select id,books,GROUP_CONCAT(talk) from tb_book GROUP BY talk;

id	books	GROUP_CONCAT(talk)
27	PHP典型模块	.net,.net
26	JAVA典型模块	JAVA,JAVA
30	aa	PHP
29	aaaa	其它

4 rows in set (0.00 sec)

mysql>

图 13.15 使用 GROUP BY 关键字与 GROUP_CONCAT()函数进行分组查询



mysql> select id,books ,user from tb_book GROUP BY user;

id	books	user
22	PHP24课堂	NULL
27	PHP典型模块	lx
28	C#项目整合	mr
25	JAVA范例完全自学手册	mrsoft

4 rows in set (0.00 sec)

mysql>

图 13.16 使用 GROUP BY 关键字实现多个字段分组

13.2.13 用 LIMIT 限制查询结果的数量

查询数据时，可能会查询出很多记录。而用户需要的记录可能只是很少一部分。这样就需要来限制查询结果的数量。LIMIT 是 MySQL 中的一个特殊关键字。LIMIT 子句可以对查询结果的记录条数进行限定，控制它输出的行数。下面通过具体实例来了解 LIMIT 的使用方法。

例 13.15 查询 tb_login 表中，按照 id 编号进行升序排列，显示前 3 条记录。**（实例位置：光盘\TM\Instance\13\13.15）**

查询语句如下：

```
select * from tb_login order by id asc limit 3;
```

查询结果如图 13.17 所示。

使用 LIMIT 还可以从查询结果的中间部分取值。首先要定义两个参数，参数 1 是开始读取的第一条记录的编号（在查询结果中，第一个结果的记录编号是 0，而不是 1）；参数 2 是要查询记录的个数。

例 13.16 查询 tb_login 表中，按照 id 编号进行升序排列，从编号 1 开始，查询两条记录。**（实例位置：光盘\TM\Instance\13\13.16）**

查询语句如下：

```
select * from tb_login where id order by id asc limit 1,2;
```

查询结果如图 13.18 所示。

mysql> select * from tb_login order by id asc limit 3;

id	user	pwd	section	name
6	lx	lx	PHP程序开发部	明日科技
7	mrkj	mrkj	程序开发	
8	mr	mrsoft	PHP	明日科技

3 rows in set (0.00 sec)

图 13.17 使用 LIMIT 关键字查询指定记录数



mysql> select * from tb_login order by id asc limit 1,2;

id	user	pwd	section	name
7	mrkj	mrkj	程序开发	高经理
8	mr	mrsoft	PHP	明日科技

2 rows in set (0.00 sec)

mysql>

图 13.18 使用 LIMIT 关键字查询指定记录

13.3 聚合函数查询

视频讲解：光盘\TM\Video\第 13 章\聚合函数查询.exe

聚合函数的最大特点是它们根据一组数据求出一个值。聚合函数的结果值只根据选定行中非 NULL 的



值进行计算，NULL 值被忽略。

13.3.1　COUNT()函数

COUNT()函数，对于除“*”以外的任何参数，返回所选择集合中非 NULL 值的行的数目；对于参数“*”，返回选择集合中所有行的数目，包含 NULL 值的行。没有 WHERE 子句的 COUNT(*)是经过内部优化的，能够快速的返回表中所有的记录总数。

例 13.17　下面使用 count()函数统计 tb_login 表中的记录数。**（实例位置：光盘\TM\Instance\13\13.17）**

查询语句如下：

```
select count(*) from tb_login;
```

查询结果如图 13.19 所示。结果显示，tb_login 表中共有 4 条记录。

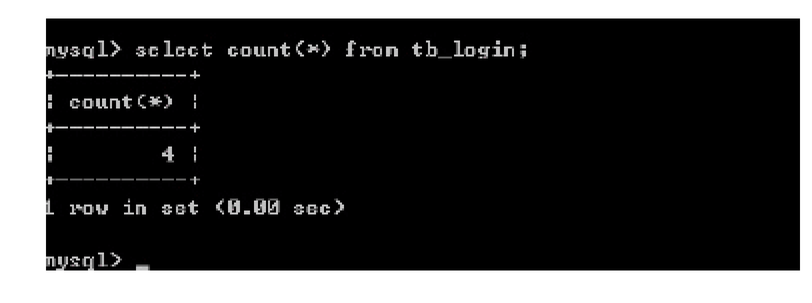


图 13.19　使用 count()函数统计记录数

13.3.2　SUM()函数

SUM()函数可以求出表中某个字段取值的总和。

例 13.18　使用 SUM()函数统计 tb_book 表中图书的访问量字段（row）的总和。在查询前，先来查询一下 tb_book 表中 row 字段的值，结果如图 13.20 所示。**（实例位置：光盘\TM\Instance\13\13.18）**

下面使用 SUM()函数来查询。查询语句如下：

```
select sum(row) from tb_book;
```

查询结果如图 13.21 所示。结果显示 row 字段的总和为 96。



图 13.20　tb_book 表中 row 字段的值



图 13.21　使用 SUM()函数查询 row 字段值的总和

13.3.3　AVG()函数

AVG()函数可以求出表中某个字段取值的平均值。

例 13.19　下面使用 AVG()函数求 tb_book 表中 row 字段值的平均值。**（实例位置：光盘\TM\Instance\13\13.19）**

查询语句如下：

```
select AVG(row) from tb_book;
```

查询结果如图 13.22 所示。

13.3.4　MAX()函数

MAX()函数可以求出表中某个字段取值的最大值。

例 13.20 下面使用 MAX()函数查询 tb_book 表中 row 字段的最大值。（**实例位置：光盘\TM\Instance\13\13.20**）

查询语句如下：

```
select MAX(row) from tb_book;
```

查询结果如图 13.23 所示。

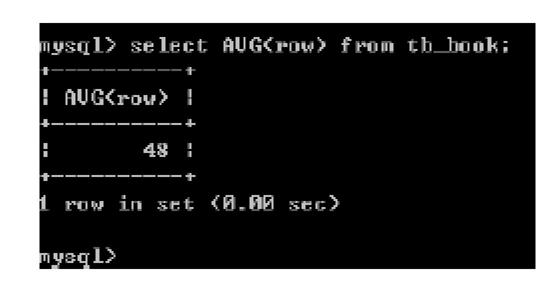


图 13.22 使用 AVG()函数求 row 字段值的平均值



图 13.23 使用 MAX()函数求 row 字段的最大值

下面来看一下 tb_book 表中 row 字段的所有值，查询结果如图 13.24 所示。结果显示 row 字段中最大值为 95，与使用 MAX 函数查询的结果一致。

13.3.5 MIN()函数

MIN()函数可以求出表中某个字段取值的最小值。

例 13.21 使用 MIN()函数查询 tb_book 表中 row 字段的最小值。（**实例位置：光盘\TM\Instance\13\13.21**）

查询语句如下：

```
select MIN(row) from tb_book;
```

查询结果如图 13.25 所示。

```
mysql> select row from tb_book;
+------+
| row  |
+------+
| 12   |
| 95   |
| NULL |
| 1    |
| 8    |
| NULL |
+------+
6 rows in set (0.00 sec)

mysql>
```

图 13.24 tb_book 表中 row 字段的所有值



图 13.25 使用 MIN()函数求 row 字段的最小值

13.4 连 接 查 询

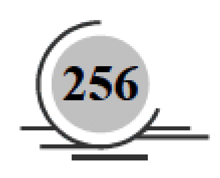
连接是把不同表的记录连到一起的最普遍的方法。一种错误的观念认为由于 MySQL 的简单性和源代码开放性，使它不擅长连接。这种观念是错误的。MySQL 从一开始就能够很好地支持连接，现在还以支持标准的 SQL2 连接语句而自夸，这种连接语句可以以多种高级方法来组合表记录。

13.4.1 内连接查询

视频讲解：光盘\TM\Video\第 13 章\内连接查询.exe

内连接是最普遍的连接类型，而且是最匀称的，因为它们要求构成连接的每一部分的每个表都匹配，不匹配的行将被排除。

内连接的最常见的例子是相等连接，也就是连接后的表中的某个字段与每个表中的都相同。这种情况下，最后的结果集只包含参加连接的表中与指定字段相符的行。



例 13.22　下面有两个表，tb_login 用户信息表和 tb_book 图书信息表，先来分别看下各表的数据，图 13.26 为 tb_login 表的数据，图 13.27 为 tb_book 表的数据。**（实例位置：光盘\TM\Instance\13\13.22）**


mysql> select * from tb_login;

id	user	pwd	section	name
7	mrkj	mrkj	程序开发	
6	lx	lx	PHP程序开发部	明日科技
8	mr	mrsoft	PHP	明日科技

3 rows in set (0.00 sec)

图 13.26　tb_login 数据表


mysql> select id,user,sort,books from tb_book;

id	user	sort	books
26	mr	典型应用类	JAVA典型模块
27	mr	模块类	PHP典型模块
28	mr	项目类	C#项目整合
29	lx	范例类	aaaa
30	lx	基础类	aa
25	mr	范例类	JAVA范例完全自学手册

6 rows in set (0.00 sec)

图 13.27　tb_book 数据表

从上面的查询结果中可以看出，在两个表中存在一个连接—— user 字段，它在两个表中是等同的，tb_login 表的 user 字段与 tb_book 表的 user 字段相等，因此可以创建两个表的一个连接。查询语句如下：

```
select name,books from tb_login,tb_book where tb_login.user=tb_book.user;
```

查询结果如图 13.28 所示。

13.4.2　外连接查询

视频讲解：光盘\TM\Video\第 13 章\外连接查询.exe

与内连接不同，外连接是指使用 OUTER JOIN 关键字将两个表连接起来。外连接生成的结果集不仅包含符合连接条件的行数据，而且还包括左表（左外连接时的表）、右表（右外连接时的表）或两边连接表（全外连接时的表）中所有的数据行。语法格式如下：

```
SELECT 字段名称 FROM 表名 1 LEFT|RIGHT JOIN 表名 2 ON 表名 1.字段名 1=表名 2.属性名 2;
```

外连接分为左外连接（LEFT JOIN）、右外连接（RIGHT JOIN）和全外连接 3 种类型。

1．左外连接

左外连接（LEFT JOIN）是指将左表中的所有数据分别与右表中的每条数据进行连接组合，返回的结果除内连接的数据外，还包括左表中不符合条件的数据，并在右表的相应列中添加 NULL 值。

例 13.23　下面使用左外连接查询 tb_login 表和 tb_book 表，通过 user 字段进行连接。**（实例位置：光盘\TM\Instance\13\13.23）**

查询语句如下：

```
select section,tb_login.user,books,row from tb_login left join tb_book on tb_login.user=tb_book.user;
```

查询结果如图 13.29 所示。


mysql> select name,books from tb_login,tb_book where tb_login.user=tb_book.user;

name	books
高经理	JAVA范例完全自学手册
明日科技	PHP典型模块
明日科技	Word范例宝典

3 rows in set (0.00 sec)

mysql>

图 13.28　内连接查询


mysql> select section,tb_login.user,books,row from tb_login left join tb_book on tb_login.user=tb_book.user;

section	user	books	row
PHP求职词典	mrkj	NULL	NULL
PHP编程词典	lx	PHP典型模块	95
PHP编程词典	lx	Word范例宝典	1
PHP开发实战词典	mrsoft	JAVA范例完全自学手册	NULL

4 rows in set (0.00 sec)

mysql>

图 13.29　左外连接查询

结果显示，第 1 条记录的 books 和 row 字段的值为空，这是因为在 tb_book 表中并不存在 user 字段为 mrkj 的值。

2．右外连接

右外连接（RIGHT JOIN）是指将右表中的所有数据分别与左表中的每条数据进行连接组合，返回的结果除内连接的数据外，还包括右表中不符合条件的数据，并在左表的相应列中添加 NULL。

例 13.24 下面使用右外连接查询 tb_book 表和 tb_login 表，两表通过 user 字段连接。**（实例位置：光盘\TM\Instance\13\13.24）**

查询语句如下：

```
select section,tb_book.user,books,row from tb_book right join tb_login on tb_book.user=tb_login.user;
```

查询结果如图 13.30 所示。

13.4.3 复合条件连接查询

视频讲解：光盘\TM\Video\第 13 章\复合条件连接查询.exe

在连接查询时，也可以增加其他的限制条件。通过多个条件的复合查询，可以使查询结果更加准确。

例 13.25 下面使用内连接查询 tb_book 表和 tb_login 表，并且 tb_book 表中 row 字段值必须大于 5。**（实例位置：光盘\TM\Instance\13\13.25）**

查询语句如下：

```
select section,tb_book.user,books,row from tb_book,tb_login where tb_book.user=tb_login.user and row>5;
```

查询结果如图 13.31 所示。



图 13.30 右外连接查询

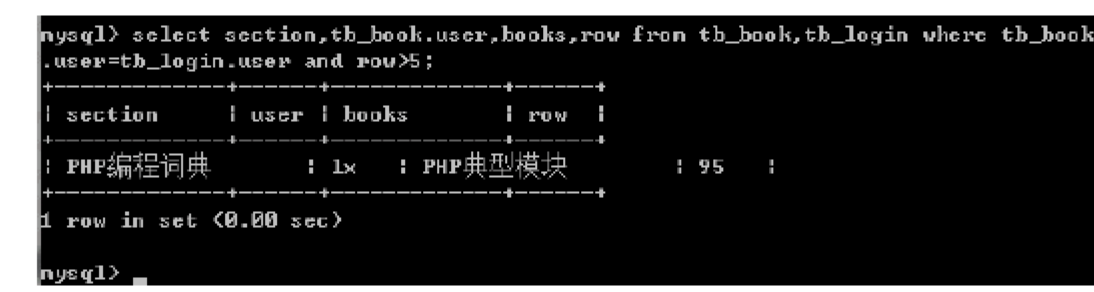


图 13.31 复合条件连接查询

13.5 子 查 询

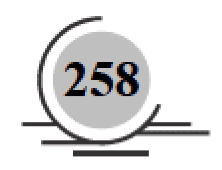
子查询就是 SELECT 查询是另一个查询的附属。MySQL 4.1 可以嵌套多个查询，在外面一层的查询中使用里面一层查询产生的结果集。这样就不是执行两个（或者多个）独立的查询，而是执行包含一个（或者多个）子查询的单独查询。

当遇到这样的多层查询时，MySQL 从最内层的查询开始，然后从它开始向外向上移动到外层（主）查询，在这个过程中每个查询产生的结果集都被赋给包围它的父查询，接着这个父查询被执行，它的结果也被指定给它的父查询。

除了结果集经常由包含一个或多个值的一列组成外，子查询和常规 SELECT 查询的执行方式一样。子查询可以用在任何可以使用表达式的地方，它必须由父查询包围，而且，如同常规的 SELECT 查询，它必须包含一个字段列表（这是一个单列列表）、一个具有一个或者多个表名字的 FROM 子句以及可选的 WHERE、HAVING 和 GROUP BY 子句。

13.5.1 带 IN 关键字的子查询

只有子查询返回的结果列包含一个值时，比较运算符才适用。假如一个子查询返回的结果集是值的列表，这时比较运算符就必须用 IN 运算符代替。



IN 运算符可以检测结果集中是否存在某个特定的值，如果检测成功就执行外部的查询。

例 13.26　下面查询 tb_login 表中的记录，但 user 字段值必须在 tb_book 表中的 user 字段中出现过。**（实例位置：光盘\TM\Instance\13\13.26）**

查询语句如下：

```
select * from tb_login where user in(select user from tb_book);
```

在查询前，先来分别看一下 tb_login 和 tb_book 表中的 user 字段值，以便进行对比，tb_login 表中的 user 字段值如图 13.32 所示，tb_book 表中的 user 字段值如图 13.33 所示。



```
mysql> select user from tb_login;
+--------+
| user   |
+--------+
| mrkj   |
| lx     |
| mrsoft |
+--------+
3 rows in set (0.00 sec)
```

图 13.32　tb_login 表中的 user 字段值



```
mysql> select user from tb_book;
+--------+
| user   |
+--------+
| mr     |
| mrsoft |
| lx     |
| lx     |
| NULL   |
| NULL   |
| NULL   |
| NULL   |
+--------+
8 rows in set (0.00 sec)
```

图 13.33　tb_book 表中的 user 字段值

从上面的查询结果可以看出，在 tb_book 表的 user 字段中没有出现 mrkj 值。下面执行带 IN 关键字的子查询语句，查询结果如图 13.34 所示。

查询结果只查询出了 user 字段值为 lx 和 mrsoft 的记录，因为在 tb_book 表的 user 字段中没有出现 mrkj 的值。

技巧　NOT IN 关键字的作用与 IN 关键字刚好相反。在本例中，如果将 IN 换为 NOT IN，则查询结果将会只显示一条 user 字段值为 mrkj 的记录。

13.5.2　带比较运算符的子查询

子查询可以使用比较运算符。这些比较运算符包括=、!=、>、>=、<、<=等。比较运算符在子查询时使用得非常广泛。

例 13.27　下面查询图书访问量为优秀的图书，在 tb_row 表中将图书访问量按访问数划分等级，如图 13.35 所示。**（实例位置：光盘\TM\Instance\13\13.27）**



```
mysql> select * from tb_login where user in(select user from tb_book);
+--------+--------+----+------------------+--------+
| user   | pwd    | id | section          | name   |
+--------+--------+----+------------------+--------+
| lx     | lx     |  6 | PHP编程词典      | 明日科技 |
| mrsoft | mrsoft |  8 | PHP开发实战词典  | 高经理  |
+--------+--------+----+------------------+--------+
2 rows in set (0.03 sec)
```

图 13.34　使用 IN 关键字实现子查询



```
mysql> select * from tb_row;
+----+-----+------+
| id | row | name |
+----+-----+------+
|  1 | 90  | 优秀 |
|  2 | 80  | 良好 |
|  3 | 70  | 一般 |
|  4 | 50  | 差   |
+----+-----+------+
4 rows in set (0.00 sec)
```

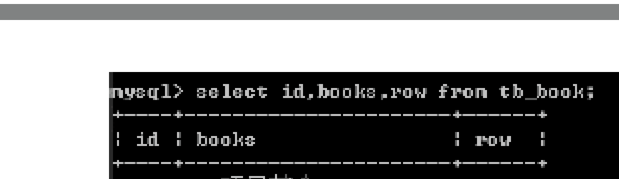
图 13.35　查询 tb_row 表中的数据

从结果中看出，当访问量大于等于 90 时即为优秀，下面再来查询 tb_book 图书信息表中 row 字段的值，如图 13.36 所示。

结果显示，第 27 条记录的访问量大于 90。下面使用比较运算符的子查询方式来查询访问量为优秀的图书信息。查询语句如下：

```
select id,books,row from tb_book where row>=(select row from tb_row where id=1);
```

查询结果如图 13.37 所示。



```
mysql> select id,books,row from tb_book;
+----+------------------------+------+
| id | books                  | row  |
+----+------------------------+------+
| 28 | C#项目整合             | NULL |
| 25 | JAVA范例完全自学手册   | NULL |
| 27 | PHP典型模块            | 95   |
| 29 | Word范例宝典           | 1    |
| 22 | PHP24课堂              | NULL |
| 23 | VB范例宝典             | NULL |
| 34 | PHP编程词典            | NULL |
| 31 | PHP网络编程自学手册    | NULL |
+----+------------------------+------+
8 rows in set (0.00 sec)
```

图 13.36　查询 tb_book 表中 row 字段的值

```
mysql> select id,books,row from tb_book where row>=(select row from tb_row where
 id=1);
+----+-------------+------+
| id | books       | row  |
+----+-------------+------+
| 27 | PHP典型模块 | 95   |
+----+-------------+------+
1 row in set (0.00 sec)

mysql> _
```

图 13.37　使用比较运算符的子查询方式来查询访问量为优秀的图书信息

13.5.3　带 EXISTS 关键字的子查询

使用 EXISTS 关键字时，内层查询语句不返回查询的记录，而是返回一个真假值。如果内层查询语句查询到满足条件的记录，就返回一个真值（true），否则，将返回一个假值（false）。当返回的值为 true 时，外层查询语句将进行查询；当返回的为 false 时，外层查询语句不进行查询或者查询不出任何记录。

例 13.28　下面使用子查询查询 tb_book 表中是否存在 id 值为 27 的记录，如果存在则查询 tb_row 表中的记录，如果不存在则不执行外层查询。（**实例位置：光盘\TM\Instance\13\13.28**）

查询语句如下：

```
select * from tb_row where exists (select * from tb_book where id=27);
```

查询结果如图 13.38 所示。

因为子查询 tb_book 表中存在 id 值为 27 的记录，即返回值为真，外层查询接收到真值后，开始执行查询。

当 EXISTS 关键与其他查询条件一起使用时，需要使用 AND 或者 OR 来连接表达式与 EXISTS 关键字。

例 13.29　如果 tb_row 表中存在 name 值为优秀的记录，则查询 tb_book 表中 row 字段大于等于 90 的记录。（**实例位置：光盘\TM\Instance\13\13.29**）

查询语句如下：

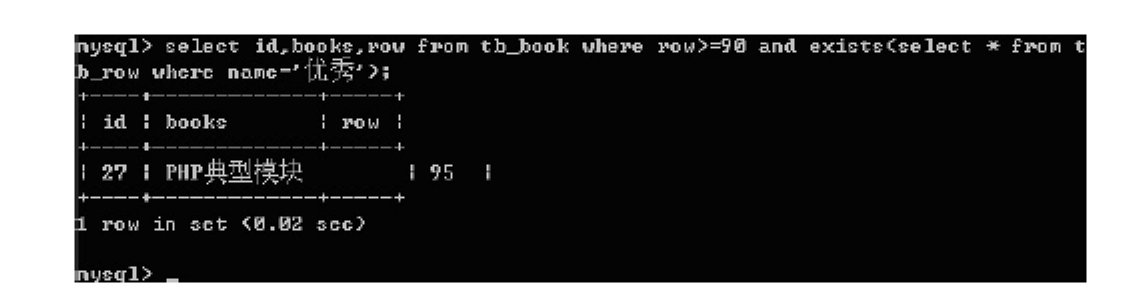
```
select id,books,row from tb_book where row>=90 and exists(select * from tb_row where name='优秀');
```

查询结果如图 13.39 所示。

```
mysql> select * from tb_row where exists(select * from tb_book where id=27);
+----+-----+------+
| id | row | name |
+----+-----+------+
|  1 | 90  | 优秀 |
|  2 | 80  | 良好 |
|  3 | 70  | 一般 |
|  4 | 50  | 差   |
+----+-----+------+
4 rows in set (0.00 sec)

mysql> _
```

图 13.38　使用 EXISTS 关键字的子查询



```
mysql> select id,books,row from tb_book where row>=90 and exists(select * from t
b_row where name='优秀');
+----+-------------+------+
| id | books       | row  |
+----+-------------+------+
| 27 | PHP典型模块 | 95   |
+----+-------------+------+
1 row in set (0.02 sec)

mysql> _
```

图 13.39　使用 EXISTS 关键字查询 tb_book 表中 row 字段大于等于 90 的记录

技巧 NOT EXISTS 与 EXISTS 刚好相反。使用 NOT EXISTS 关键字，当返回的值是 true 时，外层查询语句不执行查询；当返回值是 false 时，外层查询语句将执行查询。

13.5.4　带 ANY 关键字的子查询

ANY 关键字表示满足其中任意一个条件。使用 ANY 关键字时，只要满足内层查询语句返回的结果中的任意一个，就可以通过该条件来执行外层查询语句。

例 13.30　查询 tb_book 表中 row 字段的值小于 tb_row 表中 row 字段最小值的记录，首先查询出 tb_row 表中 row 字段的值，然后使用 ANY 关键字（“<ANY”表示小于所有值）判断。（**实例位置：光盘\TM\Instance\13\13.30**）

查询语句如下：

```
select books,row from tb_book where row<ANY(select row from tb_row);
```

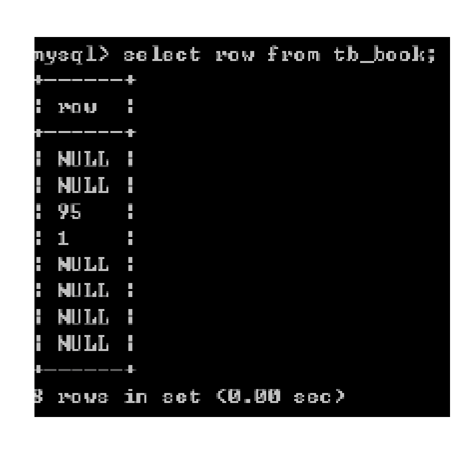


查询结果如图 13.40 所示。

```
mysql> select books,row from tb_book where row<ANY(select row from tb_row);
+----------------+------+
| books          | row  |
+----------------+------+
| Word范例宝典    | 1    |
+----------------+------+
1 row in set (0.04 sec)
```

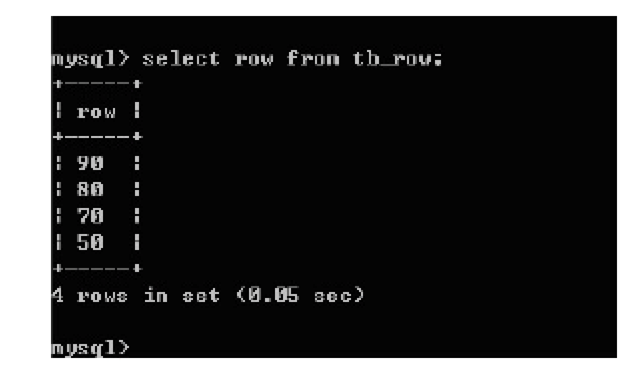
图 13.40　使用 ANY 关键字实现子查询

为了使结果更加直观，下面分别查询 tb_book 表和 tb_row 表中的 row 字段值，查询结果如图 13.41 和图 13.42 所示。



```
mysql> select row from tb_book;
+------+
| row  |
+------+
| NULL |
| NULL |
| 95   |
| 1    |
| NULL |
| NULL |
| NULL |
| NULL |
+------+
8 rows in set (0.00 sec)
```

图 13.41　tb_book 表中 row 字段的值



```
mysql> select row from tb_row;
+-----+
| row |
+-----+
| 90  |
| 80  |
| 70  |
| 50  |
+-----+
4 rows in set (0.05 sec)

mysql>
```

图 13.42　tb_row 表中 row 字段的值

结果显示，tb_row 表中 row 字段的最小值为 50，在 tb_book 表中 row 字段小于 50 的记录有 1 条，与带 ANY 关键字的子查询结果相同。

13.5.5　带 ALL 关键字的子查询

ALL 关键字表示满足所有条件。使用 ALL 关键字时，只有满足内层查询语句返回的所有结果，才可以执行外层查询语句。

例 13.31　查询 tb_book 表中 row 字段的值大于 tb_row 表中 row 字段最大值的记录，首先使用子查询，查询出 tb_row 表中 row 字段的值，然后使用 ALL 关键字（“>=ALL”表示大于等于所有值）判断。**（实例位置：光盘\TM\Instance\13\13.31）**

查询语句如下：

```
select books,row from tb_book where row>=ALL(select row from tb_row);
```

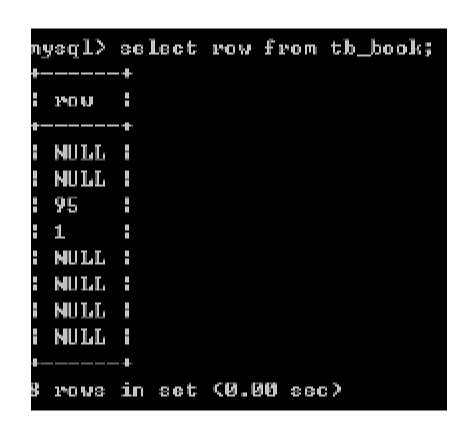
查询结果如图 13.43 所示。

```
mysql> select books,row from tb_book where row>=all(select row from tb_row);
+----------------+------+
| books          | row  |
+----------------+------+
| PHP典型模块     | 95   |
+----------------+------+
1 row in set (0.00 sec)

mysql> _
```

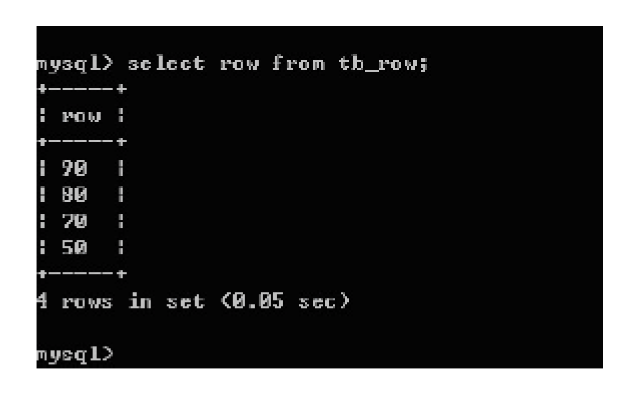
图 13.43　使用 ALL 关键字实现子查询

为了使结果更加直观，下面分别查询 tb_book 表和 tb_row 表中的 row 字段值，查询结果如图 13.44 和图 13.45 所示。



```
mysql> select row from tb_book;
+------+
| row  |
+------+
| NULL |
| NULL |
| 95   |
| 1    |
| NULL |
| NULL |
| NULL |
| NULL |
+------+
8 rows in set (0.00 sec)
```

图 13.44　tb_book 表中 row 字段的值



```
mysql> select row from tb_row;
+-----+
| row |
+-----+
| 90  |
| 80  |
| 70  |
| 50  |
+-----+
4 rows in set (0.05 sec)

mysql>
```

图 13.45　tb_row 表中 row 字段的值

结果显示，tb_row 表中 row 字段的最大值为 90，在 tb_book 表中 row 字段大于 90 的只有第 1 条记录，与带 ALL 关键字的子查询结果相同。

说明 ANY 关键字和 ALL 关键字的使用方式是一样的，但是这两者有很大的区别。使用 ANY 关键字时，只要满足内层查询语句返回的结果中的任何一个，就可以通过该条件来执行外层查询语句，而 ALL 关键字则需要满足内层查询语句返回的所有结果，才可以执行外层查询语句。

13.6 合并查询结果

合并查询结果是将多个 SELECT 语句的查询结果合并到一起。因为某种情况下，需要将几个 SELECT 语句查询出来的结果合并起来显示。合并查询结果使用 UNION 和 UNION ALL 关键字。UNION 关键字是将所有的查询结果合并到一起，然后去除相同记录；而 UNION ALL 关键字则只是简单的将结果合并到一起。下面分别介绍这两种合并方法。

例 13.32 下面查询 tb_book 表和 tb_login 表中的 user 字段，并使用 UNION 关键字合并查询结果。在执行查询操作前，先来看一下 tb_book 表和 tb_login 表中 user 字段的值，查询结果如图 13.46 和图 13.47 所示。（**实例位置：光盘\TM\Instance\13\13.32**）

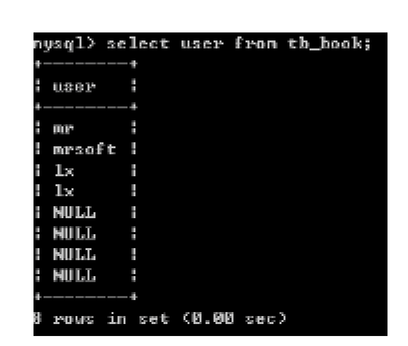


图 13.46 tb_book 表中 user 字段的值

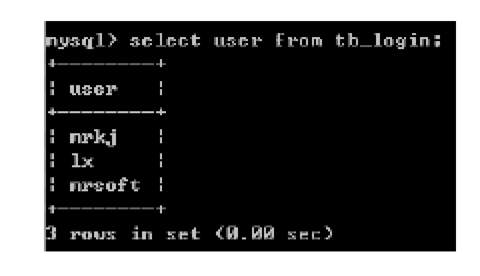


图 13.47 tb_login 表中 user 字段的值

结果显示，在 tb_book 表中 user 字段的值有 3 种，分别为 mr、mrsoft 和 lx，而 tb_login 表中 user 字段的值也有 3 种。下面使用 UNION 关键字合并两个表的查询结果，查询语句如下：

```
select user from tb_book
UNION
select user from tb_login;
```

查询结果如图 13.48 所示。结果显示，合并后将所有结果合并到了一起，并去除了重复值。

查询 tb_book 表和 tb_login 表中的 user 字段，并使用 UNION ALL 关键字合并查询结果。查询语句如下：

```
select user from tb_book
UNION ALL
select user from tb_login;
```

运行结果如图 13.49 所示。

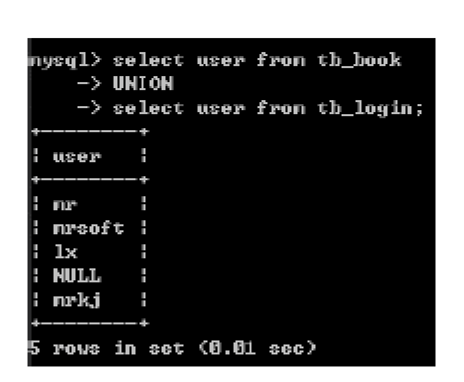


图 13.48 使用 UNION 关键字合并查询结果

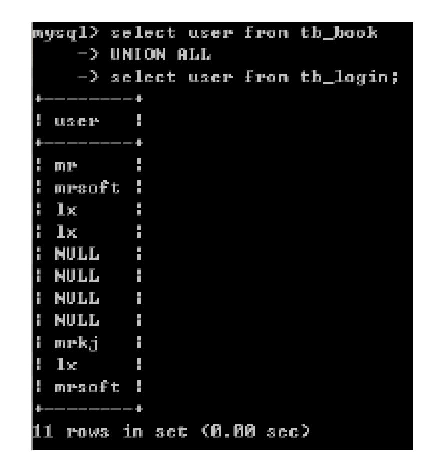


图 13.49 使用 UNION ALL 关键字合并查询结果

13.7　定义表和字段的别名

在查询时，可以为表和字段取一个别名，这个别名可以代替其指定的表和字段。为字段和表取别名，能够使查询更加方便，而且可以使查询结果以更加合理的方式显示。

13.7.1　为表取别名

当表的名称特别长时，在查询中直接使用表名很不方便。这时可以为表取一个贴切的别名。

例 13.33　下面为 tb_program 表取别名为 p，然后查询 tb_program 表中 talk 字段值为 php 的记录。（**实例位置：光盘\TM\Instance\13\13.33**）

查询语句如下：

```
select * from tb_program p where p.talk='PHP';
```

tb_program p 表示 tb_program 表的别名为 p；p.talk 表示 tb_program 表中的 talk 字段。查询结果如图 13.50 所示。

13.7.2　为字段取别名

当查询数据时，MySQL 会显示每个输出列的名词。默认情况下，显示的列名是创建表时定义的列名。我们同样可以为这个列取一个别名。

MySQL 中为字段取别名的基本形式如下：

```
字段名 [AS] 别名
```

例 13.34　下面为 tb_login 表中的 section 和 name 字段取别名，分别为 login_section 和 login_name。（**实例位置：光盘\TM\Instance\13\13.34**）

SQL 代码如下：

```
select section AS login_section,name AS login_name from tb_login;
```

查询结果如图 13.51 所示。

```
mysql> select * from tb_program p where p.talk='PHP';
+----+------+
| id | talk |
+----+------+
|  1 | PHP  |
+----+------+
1 row in set (0.00 sec)

mysql>
```

图 13.50　为表取别名



```
mysql> select section AS login_section,name AS login_name from tb_login;
+----------------+------------+
| login_section  | login_name |
+----------------+------------+
| PHP求职词典          | 明日科技       |
| PHP编程词典          | 明日科技       |
| PHP开发实战词典          | 高经理       |
+----------------+------------+
3 rows in set (0.02 sec)
```

图 13.51　为字段取别名

13.8　使用正则表达式查询

正则表达式是用某种模式去匹配一类字符串的一个方式。正则表达式的查询能力比通配字符的查询能力更强大，而且更灵活。下面详细讲解如何使用正则表达式来查询。

在 MySQL 中，使用 REGEXP 关键字来匹配查询正则表达式。其基本形式如下：

```
字段名 REGEXP '匹配方式'
```

☑ 字段名：表示需要查询的字段名称。

☑ 匹配方式：表示以哪种方式来进行匹配查询。

匹配方式参数中支持的模式匹配字符如表 13.2 所示。

表 13.2 正则表达式的模式字符

<table>
<tr><th>模式字符</th><th>含义</th><th>应用举例</th></tr>
<tr><td>^</td><td>匹配以特定字符或字符串开头的记录</td><td>使用“^”表达式查询 tb_book 表中 books 字段以字母 php 开头的记录，语句如下：
select books from tb_book where books REGEXP '^php';</td></tr>
<tr><td>$</td><td>匹配以特定符或字符串结尾的记录</td><td>使用“$”表达式查询 tb_book 表中 books 字段以“模块”结尾的记录，语句如下：
select books from tb_book where books REGEXP '模块$';</td></tr>
<tr><td>.</td><td>匹配字符串的任意一个字符，包括回车和换行</td><td>使用“.”表达式来查询 tb_book 表中 books 字段中包含 P 字符的记录，语句如下：
select books from tb_book where books REGEXP 'P.';</td></tr>
<tr><td>[字符集合]</td><td>匹配字符集合中的任意一个字符</td><td>使用“[]”表达式来查询 tb_book 表中 books 字段中包含 PCA 字符的记录，语句如下：
select books from tb_book where books REGEXP '[PCA]';</td></tr>
<tr><td>[^字符集合]</td><td>匹配除字符集合以外的任意一个字符</td><td>查询 tb_program 表中 talk 字段值中包含 c～z 字母以外的记录，语句如下：
select talk from tb_program where talk regexp '[^c-z]';</td></tr>
<tr><td>S1|S2|S3</td><td>匹配 S1、S2 和 S3 中的任意一个字符串</td><td>查询 tb_books 表中 books 字段中包含 php、c 或者 java 字符中任意一个字符的记录，语句如下：
select books from tb_books where books regexp 'php|c|java';</td></tr>
<tr><td>*</td><td>匹配多个该符号之前的字符，包括 0 和 1 个</td><td>使用“*”表达式查询 tb_book 表中 books 字段中 A 字符前出现过 J 字符的记录，语句如下：
select books from tb_book where books regexp 'J*A';</td></tr>
<tr><td>+</td><td>匹配多个该符号之前的字符，包括 1 个</td><td>使用“+”表达式来查询 tb_book 表中 books 字段中 A 字符前面至少出现过一个 J 字符，语句如下：
select books from tb_book where books regexp 'J+A';</td></tr>
<tr><td>字符串{N}</td><td>匹配字符串出现 N 次</td><td>使用{N}表达式查询 tb_book 表中 books 字段中连续出现 3 次 a 字符的记录，语句如下：
select books from tb_book where books regexp 'a{3}';</td></tr>
<tr><td>字符串{M,N}</td><td>匹配字符串出现至少 M 次，最多 N 次</td><td>使用{M,N}表达式查询 tb_book 表中 books 字段中最少出现 2 次，最多出现 4 次 a 字符的记录，语句如下：
select books from tb_book where books regexp 'a{2,4}';</td></tr>
</table>

这里的正则表达式与 Java、PHP 等编程语言中的正则表达式基本一致。

13.8.1 匹配指定字符中的任意一个

使用方括号（[]）可以将需要查询的字符组成一个字符集。只要记录中包含方括号中的任意字符，该记录将会被查询出来。例如，通过[abc]可以查询包含 a、b 和 c 等 3 个字母中任何一个的记录。

例 13.35 下面从 info 表 name 字段中查询包含 c、e 和 o 3 个字母中任意一个的记录。（**实例位置：光**

盘\TM\Instance\13\13.35）

SQL 代码如下：

```
SELECT * FROM info WHERE name REGEXP '[ceo]';
```

查询结果如图 13.52 所示。

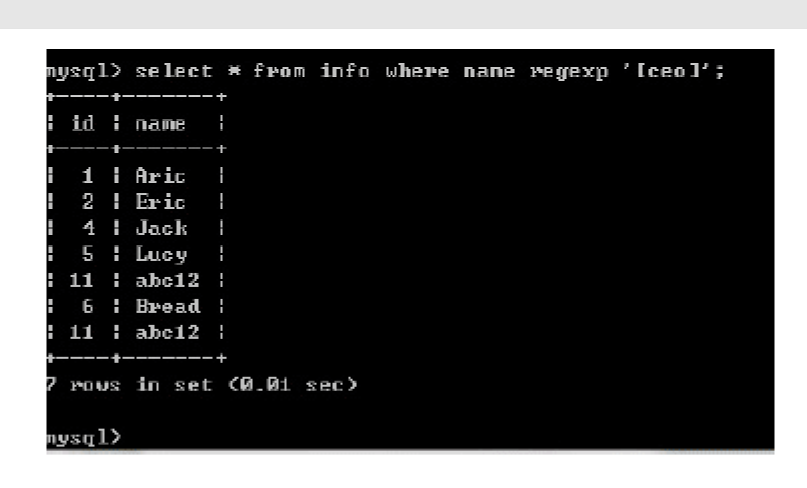


图 13.52　匹配指定字符中的任意一个

13.8.2　使用“*”和“+”来匹配多个字符

正则表达式中，“*”和“+”都可以匹配多个该符号之前的字符。但是，“+”至少表示一个字符，而“*”可以表示 0 个字符。

例 13.36　下面从 info 表 name 字段中查询字母 c 之间出现过 a 的记录。**（实例位置：光盘\TM\Instance\13\13.36）**

SQL 代码如下：

```
SELECT * FROM info WHERE name REGEXP 'a*c';
```

查询结果如图 13.53 所示。

查询结果显示，Aric、Eric 和 Lucy 中的字母 c 之前并没有 a。因为“*”可以表示 0 个，所以“a*c”表示字母 c 之前有 0 个或者多个 a 出现。上述的情况都是属于前面出现过 0 个的情况。如果使用‘+’，其 SQL 代码如下：

```
SELECT * FROM info WHERE name REGEXP 'a+c';
```

代码执行结果如图 13.54 所示。

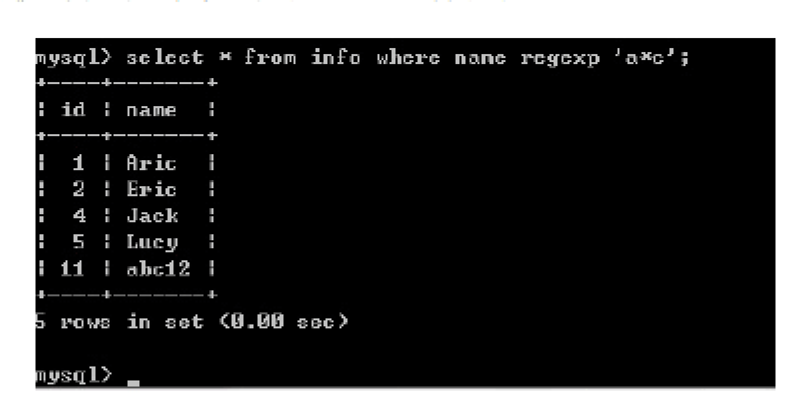


图 13.53　使用“*”来匹配多个字符



图 13.54　使用　“+”来匹配多个字符

查询结果只有一条。只有 Jack 是刚好字母 c 前面出现了 a。因为 a+c 表示字母 c 前面至少有一个字母 a。

13.9　实　　战

视频讲解：光盘\TM\Video\第 13 章\实战.exe

13.9.1　使用聚合函数 SUM 对学生成绩进行汇总

例 13.37　在学生成绩管理系统中，经常需要求出班级所有学生的各科成绩总和，从而求出所有学生的各科平均成绩。运行本实例，单击图中的“对学生成绩汇总”按钮即可将班级的各科学生成绩统计出来。**（实例位置：光盘\TM\Instance\13\13.37）**

代码如下：

```
<?php
$conn = mysql_connect("localhost","root","111") or die ("connect mysql false");        //连接 MySQL
mysql_select_db("db_database13",$conn) or die ("connect database false");          //连接 MySQL 数据库
```

```
mysql_query("set names utf8");                                        //设置编码格式
    if($_GET[page] == "1"){                                           //page 等于 1 时
    $rss = mysql_query("select *,sum(chinese+math+english) as a from tb_demo059 group by name order by
a desc ");                                                            //返回结果集
?>
<table width="580px"><tr><td background="pic/head.JPG">ID</td><td background="pic/head.JPG">学生姓名
</td><td background="pic/head.JPG">语文成绩</td><td background="pic/head.JPG">数学成绩</td><td
background="pic/head.JPG">英语成绩</td><td background="pic/head.JPG">总成绩</td></tr>
<?php
    while($rst = mysql_fetch_row($rss)){                              //循环输出结果
?>
<tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG"><?php echo
$rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background="pic/head.JPG">
<?php echo $rst[3];?></td><td background="pic/head.JPG"><?php echo $rst[4];?></td><td background=
"pic/head.JPG"><?php echo $rst[5]?></td></tr>
<?php
        }
    }
?>
```

运行结果如图 13.55 所示。



图 13.55　对学生成绩汇总

13.9.2　查询大于指定条件的记录

例 13.38　在图书管理系统中有时需要查询图书价格大于其他图书的详细信息。运行本实例，首先在图中的文本框中输入查询的图书价格应大于的数值，然后单击“查看”按钮即可将图书表中大于该数值的图书信息查找出来。（**实例位置：光盘\TM\Instance\13\13.38**）

代码如下：

```
<?php
    if($_POST['sub']){                                                //单击“查看”按钮
        if($_POST['text1'] == "" || $_POST['text1'] == "输入价格"){   //判断文本框信息
            echo "<script>alert('请输入价格');</script>";             //JavaScript 提示
        }else{
            if(preg_match("/[^A-Za-z_]/",$_POST['text1'])){           //正则表达式
                $conn = mysql_connect("localhost","root","111") or die("connect mysql false");
                mysql_select_db("db_database13",$conn) or die("connect database false");
                                                                      //连接 MySQL 数据库
                mysql_query("set names utf8");                        //设置编码格式
                $rss = mysql_query("select * from tb_demo047 where price > $_POST['text'1] order by
price");
                echo'<table width="580px"><tr><td background="pic/head.JPG">ID</td><td background=
"pic/head.JPG">书名</td><td background="pic/head.JPG">价格</td><td background="pic/head.JPG">日期
</td></tr>';
                while($rst = mysql_fetch_array($rss)){                //循环输出结果
?>

        <tr><td background="pic/head.JPG"><?php echo $rst[0];?></td><td background="pic/head.JPG">
<?php echo $rst[1];?></td><td background="pic/head.JPG"><?php echo $rst[2];?></td><td background=
"pic/head.JPG"><?php echo $rst[3];?></td></tr>
<?php
                }
```

```
                }else{
                    echo "<script>alert('不是正确的数值');</script>";        //输出提示
                }
            }
        }
?>
```

运行结果如图 13.56 所示。

图 13.56　查询大于指定条件的记录

13.9.3　使用比较运算符进行子查询

例 13.39　从 computer_stu 表中查询获得一等奖学金的学生的学号、姓名和分数。各个等级的奖学金的最低分存储在 score 表中。**（实例位置：光盘\TM\Instance\13\13.39）**

代码如下：

```
select id,name,score from computer_stu where score>=(select score from score where level=1);
```

运行结果如图 13.57 所示。

进行的子查询如图 13.58 所示。

```
mysql> select * from score;
+-------+-------+
| level | score |
+-------+-------+
|     1 |    90 |
|     2 |    80 |
|     3 |    70 |
+-------+-------+
3 rows in set (0.00 sec)

mysql> select * from computer_stu;
+------+------+-------+
| id   | name | score |
+------+------+-------+
| 1001 | Lily |    85 |
| 1002 | Tom  |    91 |
| 1003 | Jim  |    87 |
| 1004 | Aric |    77 |
| 1005 | Lucy |    65 |
| 1006 | Andy |    99 |
+------+------+-------+
6 rows in set (0.00 sec)

mysql>
```

图 13.57　全部数据



```
mysql> select id,name,score from computer_stu where score>=(select score from sc
ore where level=1);
+------+------+-------+
| id   | name | score |
+------+------+-------+
| 1002 | Tom  |    91 |
| 1006 | Andy |    99 |
+------+------+-------+
2 rows in set (0.00 sec)

mysql> _
```

图 13.58　使用比较运算符进行查询

13.9.4　GROUP BY 与 HAVING 关键字

例 13.40　下面按 tb_student 表的 sex 字段进行分组查询，然后显示记录数大于等于 3 的分组。**（实例位置：光盘\TM\Instance\13\13.40）**

代码如下：

```
SELECT sex,COUNT(sex) FROM tb_student GROUP BY sex HAVING COUNT(sex)>=3;
```

运行结果如图 13.59 所示。

13.9.5　用符号“.”来替代字符串中的任意一个字符

例 13.41　下面使用符号“.”来替代字符串中的任意一个字符。**（实例位置：光盘\TM\Instance\13\13.41）**

代码如下：

```
SELECT * FROM aa WHERE name REGEXP '^L..y$';
```

运行结果如图 13.60 所示。

```
mysql> select sex,count(sex) from tb_student group by sex having count(sex)>=3;
+------+------------+
| sex  | count(sex) |
+------+------------+
| 女   |          3 |
| 男   |          6 |
+------+------------+
2 rows in set (0.10 sec)

mysql>
```

图 13.59　GROUP BY 与 HAVING 关键字



图 13.60　用符号“.”来替代字符串中任意一个字符

13.10　小　　结

本章对 MySQL 数据库常见的查询方法进行了详细讲解，并通过大量的举例说明，使读者更好地理解所学知识的用法。在阅读本章时，读者应该重点掌握多条件查询、连接查询、子查询和查询结果排序，及使用正则表达式来查询。正则表达式的功能很强大，使用起来很灵活。希望读者能够阅读有关正则表达式的书籍，使自己对正则表达式了解得更加透彻。

13.11　学习成果检验

1．如何查询所有字段？（**答案位置：光盘\TM\Instance\13\13.42**）
2．如何查询带 like 的字符匹配查询？（**答案位置：光盘\TM\Instance\13\13.43**）
3．如何为表取别名？（**答案位置：光盘\TM\Instance\13\13.44**）
4．什么是子查询？（**答案位置：光盘\TM\Instance\13\13.45**）

第14章

综合实例（二）——留言本

（视频讲解：35 分钟）

以构建一个适合于中小型企业应用的留言本为目的，全面介绍留言本的设计思路及流程。在介绍留言本开发过程的同时，对 PHP 中一些常用函数及编程技巧也会详细介绍。通过本章的讲解，不仅可以对留言本的开发过程及思路有个全面地掌握，而且还可以学习到许多 PHP 的编程技巧，从而全面提高个人的基础及编程能力。留言本的开发过程虽然较为简单，却涵盖了动态网站开发的大部分经典功能模块，所以只要熟悉掌握本章的内容，并且能够灵活应用，就可以开发出大型的网上论坛、博客以及其他一些网络中较流行的网站。

通过阅读本章内容，你可以：

- 了解留言本的设计思路及流程
- 掌握 MySQL 数据库的设计方式和方法
- 掌握 PHP 连接 MySQL 数据库的方法
- 熟悉运用常见的 SQL 查询语句
- 掌握 PHP 实现数据分页显示的方法
- 熟悉对留言本的添加、查看和编辑

14.1　留言本概述

网络上随处可见各种各样的留言板，一个网页甚至一条新闻都有一个留言板支持。而留言本是留言板的一个纵向延伸，留言本除具有留言、查看、回复、查询、删除等功能外，还增加了管理员（也称为版主）管理功能。留言本被广泛应用于小规模企业，它不仅为企业员工提供一个交流的平台，也成为企业与员工之间交流的纽带。

14.2　系统分析流程

视频讲解：光盘\TM\Video\第 14 章\系统分析流程.exe

14.2.1　程序业务流程

用户在使用留言本进行留言时，首先应该注册，注册成功后即可登录本站，在登录本留言本时，系统会判断用户的级别，并根据用户的级别为其分配不同的权限，非本留言本的用户只具有查看留言内容的权限，普通注册用户具有添加留言、编辑个人留言和删除个人发布的留言以及回复其他留言的权限，而管理员除具有上述权限外，还具有删除全部留言（即留言和回复）的权限以及对自己发表的留言进行回复的权限。用户对本留言本进行操作的业务流程如图 14.1 所示。

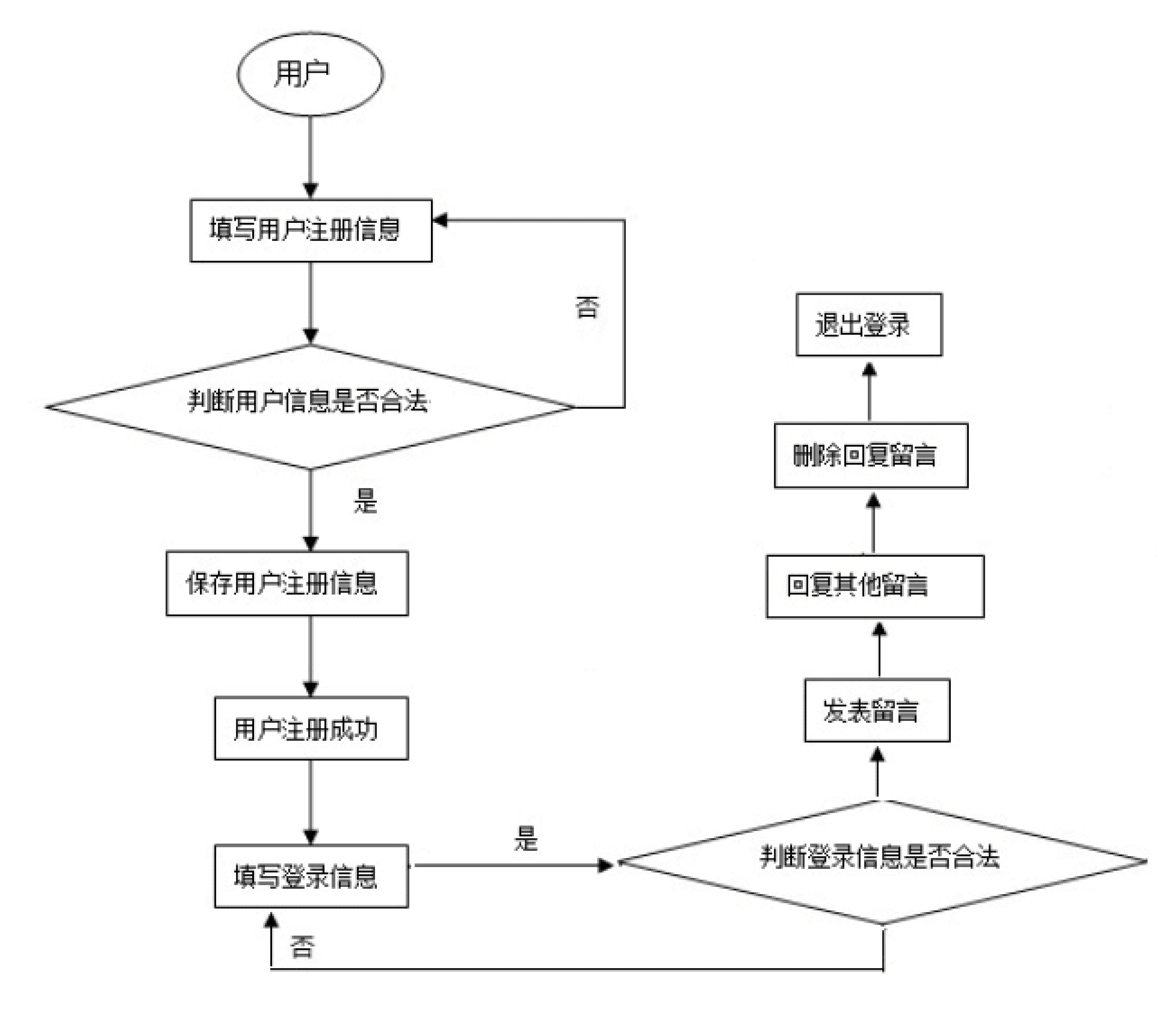
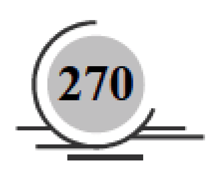


图 14.1　留言本业务流程图

14.2.2　系统预览

留言本由多个板块组成，为了让读者对本留言本有个初步的了解和认识，下面列出几个典型功能的页



面，其他页面参见光盘中的源程序。

用户注册信息界面如图 14.2 所示，该页面显示了游客注册时需要填写的昵称、密码和 E-mail 地址等相关内容。



图 14.2　用户注册信息界面

发表留言的运行效果如图 14.3 所示，该页面用于注册后用户发表留言，包括留言主题和留言内容。

查询页面运行效果如图 14.4 所示，该页显示执行查询后所查到的留言主题、留言内容，以及相关回复。

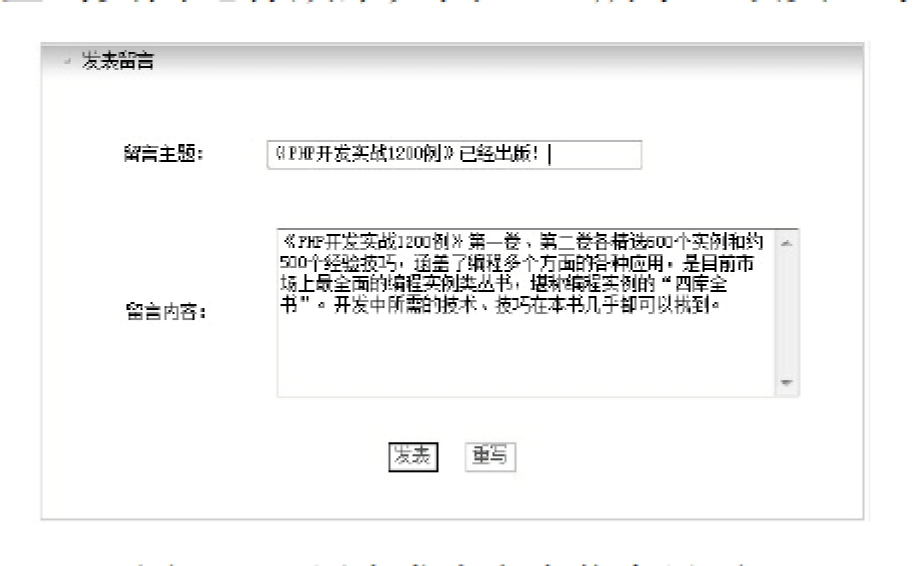


图 14.3　用户发表留言信息界面

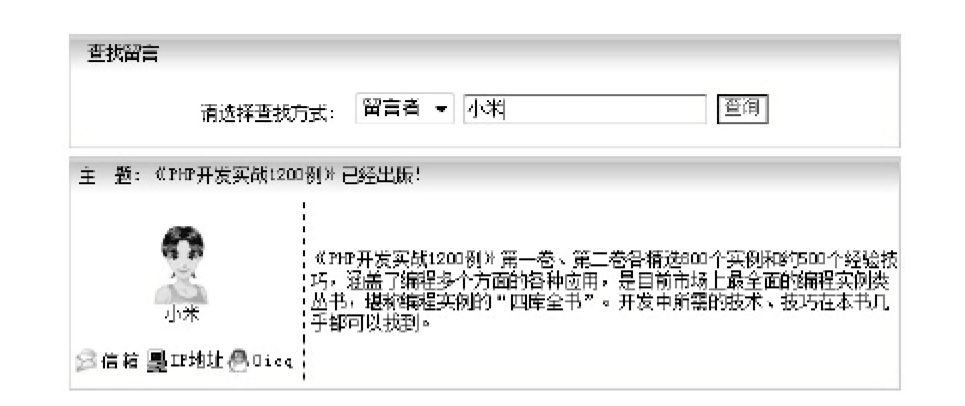


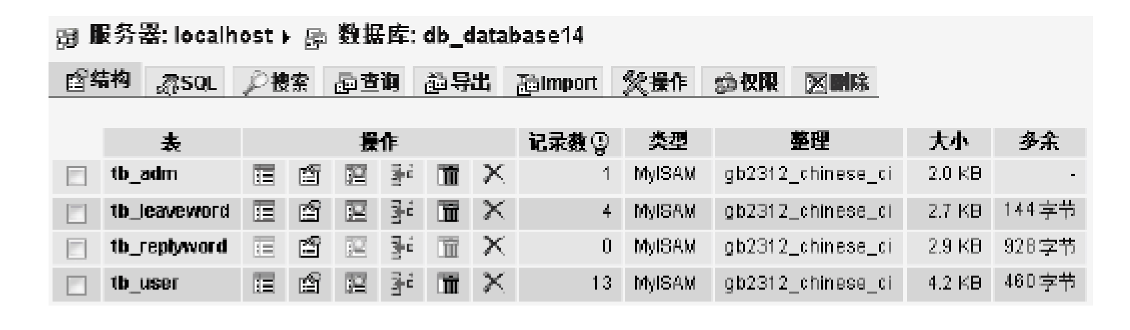
图 14.4　查询界面

14.3　数据库设计

视频讲解：光盘\TM\Video\第 14 章\数据库设计.exe

14.3.1　数据库概要说明

在留言本中，采用的是 MySQL 数据库，用来存储注册用户的基本信息、留言信息、回复信息和管理员信息等。这里将数据库命名为 db_database14，其中包含的数据表如图 14.5 所示。



服务器: localhost ▸ 数据库: db_database14

结构　SQL　搜索　查询　导出　Import　操作　权限　删除

表	记录数	类型	整理	大小	多余
tb_adm	1	MyISAM	gb2312_chinese_ci	2.0 KB	-
tb_leaveword	4	MyISAM	gb2312_chinese_ci	2.7 KB	144 字节
tb_replyword	0	MyISAM	gb2312_chinese_ci	2.9 KB	928 字节
tb_user	13	MyISAM	gb2312_chinese_ci	4.2 KB	460 字节

图 14.5　数据库结构

14.3.2　数据库概念设计

根据系统分析和功能上的需要，规划出留言本中数据表的实体 E-R 图，包括用户信息实体、留言信息实体、回复信息实体和管理员（版主）信息实体。

用户信息实体包括用户名、密码、邮箱、头像等多项资料，用户信息实体 E-R 图如图 14.6 所示。

留言信息实体包括留言者、留言主题、留言内容、留言时间等。留言信息实体 E-R 图如图 14.7 所示。

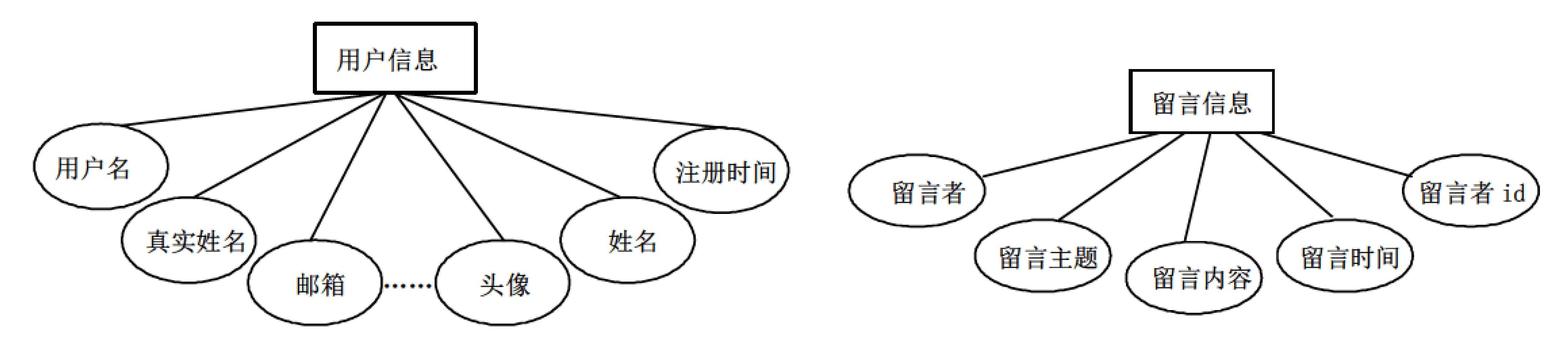


图 14.6　用户信息实体 E-R 图　　　图 14.7　留言信息实体 E-R 图

回复信息实体包括回复者、回复主题、回复内容、回复时间等。回复信息实体 E-R 图如图 14.8 所示。

管理员（版主）信息实体包括版主名称、版主登录密码、版主邮箱等。版主信息实体 E-R 图如图 14.9 所示。

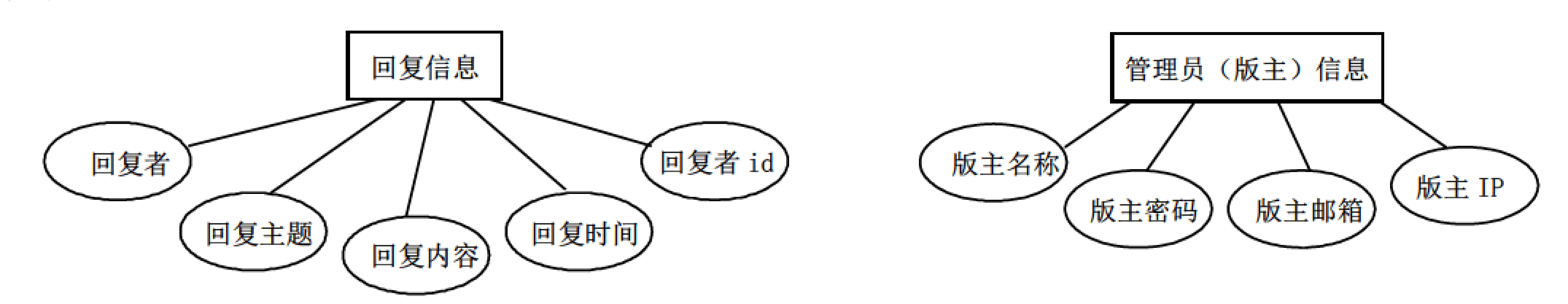


图 14.8　回复信息实体 E-R 图　　　图 14.9　版主信息实体 E-R 图

14.3.3　数据库逻辑设计

根据设计好的 E-R 图在数据库中创建数据表，下面给出重要的数据表结构。

☑　tb_user（用户信息表）

用户信息表用于存储用户的相关信息，用户信息表的结构如表 14.1 所示。

表 14.1　tb_user 表的结构

字 段 名 称	数 据 类 型	字 段 大 小	是否主键	说　　明
Id	int	4	主键	自动编号 id
Usernc	varchar	50		用户名
Userpwd	varchar	50		用户密码
Truename	varchar	50		用户真实姓名
Email	varchar	50		用户 E-mail 地址
Qq	varchar	20		用户 QQ 号码



续表

字段名称	数据类型	字段大小	是否主键	说　明
Tel	varchar	20		用户联系电话
Ip	varchar	2		用户注册 IP
Address	varchar	250		用户联系地址
Face	varchar	50		用户头像
Regtime	datetime			用户注册时间
Sex	varchar	2		用户性别
Usertype	int	2		用户类型标记

☑　tb_leaveword（留言信息表）

留言信息表存储了用户留言的相关信息，留言信息表的结构如表 14.2 所示。

表 14.2　tb_leaveword 表的结构

字段名称	数据类型	字段大小	是否主键	说　明
Id	int	8	主键	自动编号
Userid	int	8		用户 id
Createtime	datetime			发表留言时间
Title	varchar	250		留言主题
Content	text			留言内容

☑　tb_replyword（回复信息表）

回复留言信息表用于存储用户回复留言的信息，回复留言信息表的结构如表 14.3 所示。

表 14.3　tb_replyword 表的结构

字段名称	数据类型	字段大小	是否主键	说　明
Id	int	4	主键	自动编号 id
Userid	int	4		用户留言 id
Createtimes	datetime			回复时间
Titles	varchar	100		回复主题
Contents	text			回复内容
Leave_id	int	4		回复者 id

☑　tb_adm（管理员（版主）信息表）

管理员信息表用于存储管理员的信息，管理员信息表的结构如表 14.4 所示。

表 14.4　tb_adm 表的结构

字段名称	数据类型	字段大小	是否主键	说　明
Id	Int	5	主键	自动编号 id
Userword	varchar	20		版主名称
password	varchar	20		版主密码
Email	varchar	30		版主 E-mail 地址
Ip	varchar	10		版主 ip

14.4 公共模块设计

视频讲解：光盘\TM\Video\第 14 章\公共模块设计.exe

14.4.1 数据库连接文件

在进行程序开发的过程中，有很多的地方涉及到数据库的应用，在应用数据库前首先要与数据库建立连接，因此可以将数据库的连接代码作为一个公共文件进行存储，在需要使用数据库连接文件的地方直接调用该文件即可，无需重复编写相同的代码，既减少代码的冗余，也便于对数据库连接文件进行修改。在本项目中将数据库的连接代码存储于根目录下 conn 文件夹的 conn.php 文件中。

（代码位置：光盘\TM\Instance\14\conn\conn.php）

```
<?php
$conn=mysql_connect("localhost","root","111");            //连接数据库服务器
mysql_select_db("db_database14",$conn);                    //连接指定的数据库
mysql_query("set names gb2312");                           //对数据库中的编码格式进行转化，避免出现中文乱码
?>
```

注意 由于本机的 MySQL 服务器的用户名为 root，密码为 111。读者运行本程序时，需要在 conn.php 文件中对 MySQL 服务器的用户名和密码进行修改，以保证程序的正常运行。

成功创建数据库 conn.php 文件后，如果某个页面中需要进行数据库的操作，在页面中直接通过 include 语句包含 conn.php 文件即可。其代码如下：

```
<?php
     include("conn/conn.php");
?>
```

14.4.2 将文本中的字符转换为 HTML 标识符

在输出数据库中数据的过程中，有必要将数据中的一些特殊字符转换为 HTML 标识符，这样可以避免一些不必要的麻烦。例如，在输出一个程序的执行代码的过程中，如果不对其进行转换，那么输出的将不是程序的代码，而是程序的执行结果。这里将文本中字符的转换编写到一个自定义函数 unhtml()中，保存于 function.php 中，将其作为一个公共模块来使用，当需要使用时直接调用 function.php 文件即可。

function.php 文件包含两个自定义函数：unhtml()和 msubstr()。unhtml()函数用于将数据中的特殊字符转换为 HTML 标识符；msubstr()函数用于对字符串进行指定长度的截取。

（代码位置：光盘\TM\Instance\14\function.php）

```
<?php
function unhtml($content) {                                   //定义自定义函数的名称
❶           $content=htmlspecialchars($content);               //转换文本中的特殊字符
❷           $content=str_replace("@","",$content);             //替换文本中的换行符
❸            return trim($content);                            //删除文本中首尾的空格
 }                                 //定义一个用于截取一段字符串的函数 msubstr()
 function msubstr($str,$start,$len) {  //$str 指的是字符串，$start 指的是字符串的起始位置，$len 指的是长度
     $strlen=$start+$len;              //用$strlen 存储字符串的总长度（从字符串的起始位置到字符串的总长度）
```

```
        $tmpstr="";                          //声明一个变量，并赋值为空
    for($i=0;$i<$strlen;$i++) {              //通过 for 循环语句读取字符串
        if(ord(substr($str,$i,1))>0xa0) {    //如果字符串首个字节的 ASCII 序数值大于 0xa0，则表示汉字
            $tmpstr.=substr($str,$i,2);      //每次取出两位字符赋给变量$tmpstr，即等于一个汉字
            $i++;                            //变量自加 1
        } else                               //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
❹              $tmpstr.=substr($str,$i,1);
    }
    return $tmpstr;                          //输出字符串
}
?>
```

说明

❶htmlspecialchars()：将特殊字符转换成 HTML 格式，而不会将所有字符都转换成 HTML。

❷str_replace()：将所有在参数 subject 中出现的 search 以参数 replace 替换，参数&count 表示替换字符串执行的次数。

❸trim()：删除字符串中首尾的空白或者其他字符。

❹substr()：从指定的字符串中按照指定的位置截取一定长度的字符。

14.4.3　JavaScript 脚本

在留言本中，JavaScript 脚本一般用于表单元素验证，如判断 text 文本框输入是否为空，输入格式是否符合标准等。在网页中使用 JavaScript 脚本的方式主要有 3 种方式。

（1）在网页中使用<script></script>标签对

<script></script>标签对可以放在网页的任意位置，一般是放在<head></head>或<body></body>之间。输出系统当前时间的 JavaScript 脚本代码如下：

```
<script language=JavaScript>
    today=new Date();
    function initArray(){
    this.length=initArray.arguments.length
    for(var i=0;i<this.length;i++)
    this[i+1]=initArray.arguments[i]    }

    var d=new initArray(
      "星期日",
      "星期一",
      "星期二",
      "星期三",
      "星期四",
      "星期五",
      "星期六");
document.write(
      "<font color=#000000 style='font-size:9pt;font-family: 宋体'> ",
      today.getYear(),"年",
      today.getMonth()+1,"月",
      today.getDate(),"日",
        "  ",
      d[today.getDay()+1],
      "</font>" );
</script>
```

（2）在单独文件中使用

如果 JavaScript 脚本比较多，而且位置分散不易管理，可以统一放到一个扩展名为 js 的文件中，使该文

件成为 JavaScript 脚本文件。在脚本文件中，不需要使用<script></script>标签对，直接写脚本代码即可。当页面需要使用到里面的 JavaScript 脚本时，可以这样引用，代码如下：

```
<SCRIPT src="calendar.js"></SCRIPT>
<SCRIPT lnguage="Javascript">
var xhnetCalendar=new calendar("xhnetCalendar","xhnetCal");
</SCRIPT>
<SCRIPT> xhnetCalendar.channel='forum';</SCRIPT>
<DIV class=calendar id=abc>
<SCRIPT>
document.write(xhnetCalendar.generate(null));
xhnetCalendar.focusDate();
</SCRIPT>
```

注意 本留言本主页上的万年历就是采用这种方式引用的。万年历的主要代码请参见光盘的 calendar.js 文件。

（3）在表单元素或标签中使用

这是最直接的使用方式，如果是少量的脚本则可以这样使用。例如，在超链接标签<a>中想要使用 JavaScript 脚本，代码格式如下：

```
<a href="reply.php?t_id=<?php echo $_GET['l_id']; ?>&loor=<?php echo $i;?>">回复</a>
```

在留言本中，这 3 种方法都有使用到，在后面涉及到具体应用时再进行说明。

14.5　首页模块设计

视频讲解：光盘\TM\Video\第 14 章\首页模块设计.exe

14.5.1　首页设计概述

本系统首页页面设计得比较简洁明了，主要包括以下 4 部分内容。

- ☑ 首部导航栏：包括首页链接、用户注册、发表留言、查看留言、查询留言、版主管理和注销登录模块。
- ☑ 左侧显示区：包括用户登录、万年历和最新留言模块。游客通过该区域登录留言本，以及了解最新的留言主题。
- ☑ 主显示区：为留言本的最新留言。游客通过该区域可以查看最新的留言主题与留言内容。
- ☑ 尾部显示区：为留言本的版权声明。

留言本首页效果图如图 14.10 所示。



图 14.10　主界面

14.5.2　switch 和 include 语句

留言本首页设计主要应用 switch 语句和 include 包含语句，其实现的原理是：应用 switch 语句，根据超链接中传递的变量值进行判断，根据不同的变量值应用 include 调用不同的子文件。

为了更好地理解这个技术，先来了解一下 switch 语句。该语句的格式如下：

```
switch(expr){                                  //expr 条件为变量名称
    case expr1:                                //case 后的 expr1 为变量的值
        statement1;                            //冒号（:）后的是符合该条件时要执行的部分
    break;                                     //应用 break 来跳离循环体
    case expr2:
        statement2;
    break;
    default:
        statementN;
    break;
}
```

参数 expr 是表达式的值，即 switch 语句的条件变量的名称；参数 expr1 放置于 case 语句后，是要与条件变量 expr 进行匹配的值中的一个；statement1 是参数 expr1 的值与条件变量 expr 的值相匹配时执行的代码；break 语句实现终止语句的执行，即当语句在执行过程中，遇到 break 就停止执行，跳出循环体；default 是 case 的一个特例，匹配了任何其他 case 都不匹配的情况，并且是最后一条 case 语句。

通过 switch 和 include 语句来实现首页的设计是一个很好的方法，不但实现过程简单，而且操作非常灵活。其关键代码如下：

（代码位置：光盘\TM\Instance\14\index.php）

```
<?php
    switch($id){                               //根据变量提交的不同值
        case "首页":                           //判断与变量提交的值是否相同
            include "main.php";                //如果值相同，则调用指定文件
        break;                                 //并且跳出本次循环
        case "用户注册":
            include "reg.php";
        break;
        case "发表留言":
            include "leaveword.php";
        break;
        case "查看留言":
            include "lookleaveword.php";
        break;
        case "查询留言":
            include "searchword.php";
        break;
        case "版主管理":
            include "login.php";
        break;
        case "注销登录":
            include "logout.php";
        break;
        case "编辑留言":
            include "editleaveword.php";
        break;
        case "回复编辑留言":
```

```
			include "edlitreplyword.php";
		break;
		case "详细信息":
			include "lookxx.php";
		break;
		default:								//判断当该值等于空时，调用该文件
			include "main.php";
		break;
		}
	?>
```

14.5.3 首页实现过程

在一个网站中，首页被访问的次数是比较多的。为了加快页面的运行速度、提高访问量，本项目首页使用 include 语句包含主要功能模块。其代码如下：

（代码位置：光盘\TM\Instance\14\index.php）

```
<?php
❶	include_once("top.php");
?>
<table width="779" height="23" border="0" align="center" cellpadding="0" cellspacing="0">
  <tr>
    <td width="292" background="images/dh_back.gif"><div align="right">今天是： 
<script language=JavaScript>
   today=new Date();
   function initArray(){
   this.length=initArray.arguments.length
   for(var i=0;i<this.length;i++)
   this[i+1]=initArray.arguments[i]   }
   var d=new initArray(
     "星期日",
     "星期一",
     "星期二",
     "星期三",
     "星期四",
     "星期五",
     "星期六");
document.write(
     "<font color=#000000 style='font-size:9pt;font-family: 宋体'> ",
     today.getYear(),"年",
     today.getMonth()+1,"月",
     today.getDate(),"日",
       "  ",
     d[today.getDay()+1],
     "</font>" );
</script></div></td>
❷	<td width="200" valign="top"><?php include_once("left.php");?></td>
	</td>
❸	 <!--留言信息-->
   </tr>
</table>
<?php
❹	include_once("bottom.php");
?>
```



说明

❶应用 include 语句包含 top.php 文件，该文件用于显示网站导航、留言本名称及当前登录的用户名称。

❷应用 include 语句包含 left.php 文件，该文件用于显示用户登录、万年历及最新留言信息。

❸在首页（index.php）中，应用表格布局的方式显示留言内容。

❹应用 include 语句包含 bottom.php 文件，该文件用于显示版权信息。

14.6　用户注册模块设计

视频讲解：光盘\TM\Video\第 14 章\用户注册模块设计.exe

14.6.1　用户注册模块概述

留言本为了更好地与用户进行交流和沟通，创建了一个用户注册模块，通过用户注册模块，可以有效地对用户信息进行采集，并将合法的用户信息保存到指定的数据表中，实现与用户的长期沟通和交流。用户注册模块的运行结果如图 14.11 所示。

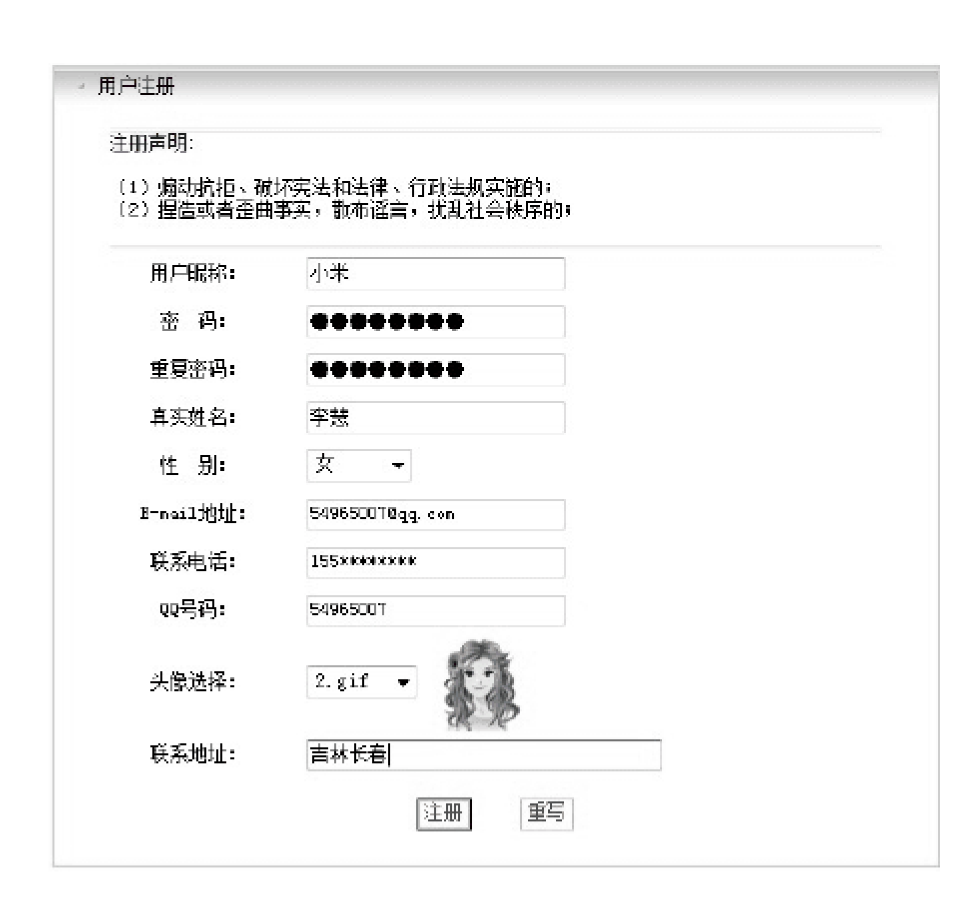


图 14.11　用户注册模块的运行结果

14.6.2　JavaScript 脚本验证表单元素

在用户注册模块中，必不可少的就是要对用户输入的信息进行判断，首先判断用户填写的注册信息中哪些是必须填写的，哪些可以不填写，然后进一步判断输入的信息是否合理合法，如判断输入的邮箱的格式是否正确、电话号码是否有效等。对表单中提交的数据进行判断最常用的办法就是使用 JavaScript 脚本，也可以使用正则表达式。下面讲解在本模块中是如何通过 JavaScript 脚本实现表单提交数据的验证。

操作原理是：在 form 表单中调用 onSubmit 事件，通过该事件调用指定的 JavaScript 脚本，执行自定义函数 chkinput()，实现对表单中提交数据的验证。在 JavaScript 脚本中，实现对表单中提交数据的判断，判断输入的内容是否为空、内容格式是否正确，如果正确则继续执行，否则将弹出提示对话框，并将鼠标的焦点指定到出错的位置。具体的 JavaScript 脚本代码如下：

（代码位置：光盘\TM\Instance\14\reg.php）

```
<script language="javascript">
        function chkinput(form){                        //定义一个函数
            if(form.usernc.value==""){                  //判断 usernc 文本框中的值是否为空
                alert("请输入用户昵称！");              //如果为空则输出“请输入用户昵称”
                form.usernc.focus();                    //返回到 tel 文本框
                return(false);
            }
```

```
                    if(form.userpwd.value==""){
                        alert("请输入注册密码！ ");
                      form.userpwd.focus();
                      return(false);
                     }
                   if(form.userpwd1.value==""){
                        alert("请输入重复密码！ ");
                      form.userpwd1.focus();
                      return(false);
                   }
                    if(form.userpwd.value!=form.userpwd1.value){

                      alert("密码与确认密码不同！ ");
                      form.userpwd.focus();
                      return(false);
                   }
               var i=form.email.value.indexOf("@");
                 var j=form.email.value.indexOf(".");
                 if((i<0)||(i-j>0)||(j<0)){
                   alert("请输入正确的 E-mail 地址!");
                    form.email.select();
                    return(false);
                 }
                return(true);                                    //提交表单

               }
</script>
```

上述代码中，只是列举了 JavaScript 中的部分内容，并且在对电话号码进行判断时，只是判断其是是否为数字，没有进一步判断电话号码的格式是否正确。如果想要更加准确地判断电话号码的格式是否正确可以采用下面的方法。

通过正则表达式的 preg_match()函数，在表单提交处理页中对电话号码进行判断。preg_match()函数的语法格式如下：

```
int preg_match ( string pattern, string subject [, array matches [, int flags]] )
```

preg_match()函数的参数说明如表 14.5 所示。

表 14.5　preg_match()函数的参数说明

参　数	说　明
pattern	必选参数。需要匹配的正则表达式
subject	必选参数。输入的字符串
matches	可选参数。输出搜索结果的数组，如$out[0]将包含与整个模式匹配的结果，$out[1]将包含与第一个捕获的括号中的子模式所匹配的结果，依次类推
flags	可选参数。若设为 PREG_OFFSET_CAPTURE，对每个出现的匹配结果也同时返回其附属的字符串偏移量，本标记自 PHP 4.3.0 起可用

通过 preg_match()函数判断电话号码的格式是否正确的方法如下。

首先定义一个用于判断电话号码格式的正则表达式。代码如下：

```
/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$/
```

正则表达式的功能分析如下：使用“^”和“$”对字符串进行边界的限制，对区号从字符串的开始进行匹配，对其他号码从字符串的末尾开始进行匹配；将括号“()”中的内容作为一个原子使用；使用“\d”来



匹配一个数字，区号为 3 或 4 个数字，其他数字为 7 或 8 个；使用“{}”来对前字符进行重复匹配；使用“|”对匹配的模式进行选择，分成 3 个模式。

然后将该正则表达式应用到 preg_match()函数中，对表单提交的电话号码进行判断，如果正确则继续执行，否则弹出提示信息，并返回到表单提交页。代码如下：

```
<?php
   $tel="0431-84978981";                                              //定义一个电话号码的变量
   if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$tel,$counts)){
        echo "您输入的电话号码格式正确!";                              //输出字符串
   }else{
        echo "<script>alert('您输入的电话号码的格式不正确!!');history.back()</script>";
   }
   ?>
```

14.6.3　用户注册模块实现过程

注册模块的实现过程非常简单，首先阅读注册声明，然后填写用户注册的用户名和密码，提交后由系统判断输入的用户名是否被占用，如果未被占用则可以继续注册，将数据提交到表单处理页进行处理，最后将用户注册的信息保存到指定的数据表中。

用户注册模块由两个文件组成：reg.php 文件用于填写详细的注册信息；savereg.php 文件用于对表单中提交的数据进行处理，将数据保存到指定的数据表中。

在 savereg.php 文件中，首先连接数据库，然后获取到表单中提交的数据，并且判断提交的用户名是否被占用，最后将提交的数据进行处理，并将数据保存到指定的数据表中。程序代码如下：

（代码位置：光盘\TM\Instance\14\savereg.php）

```
<?php
session_start();                                       //启用 session 支持
if(isset($_SESSION['userword'])){                      //获取登录的版主名称
echo "<script>alert('在同一台机器上，不允许同时使用用户名和管理员进行登录！');
window.location.href='index.php';</script>";        //给出用户与版主不能同时登录的信息
     }else{
 include_once("conn/conn.php");                        //包含数据库文件
 $usernc=$_POST["usernc"];                             //调用注册时提交的用户名称
 $sql=mysql_query("select usernc from tb_user where usernc='".$usernc."'",$conn);
                                                       //查询用户表中已注册用户的名称是否与当前提交的用户名相同
 $info=mysql_fetch_array($sql);                        //输出查询的数据
          if($info){
   echo "<script>alert('对不起，你的昵称已经被占用！');history.back();</script>";          //相同的给出提示
   exit;
                  }
 $ip=$_SERVER["REMOTE_ADDR"];
 if(mysql_query("insert into tb_user(usernc,userpwd,truename,email,qq,tel,ip,address,face,regtime,sex,usertype)
values('".$usernc."','".md5(trim($_POST["userpwd"]))."','".$_POST["truename"]."','".$_POST["email"]."','".$_POS
T["qq"]."','".$_POST["tel"]."','".$ip."','".$_POST["address"]."','".$_POST["face"]."','".date("Y-m-d
H:i:s")."','".$_POST["sex"]."','0')",$conn))
 {
   $_SESSION["unc"]=$usernc;
   echo "<script>alert('注册成功！');window.location.href='index.php'</script>";
 }else{
   echo "<script>alert('注册失败！');history.back();</script>";
       }
}
?>
```

14.7 添加留言模块设计

14.7.1 添加留言模块概述

当用户登录后，单击首页上的“发表留言”超链接，将加载发表留言表单。用户填写完留言标题、留言内容后，单击“发表”按钮，跳转到留言处理页进行处理。用户添加的留言将会保存到 MySQL 数据库中。添加留言的界面如图 14.12 所示。

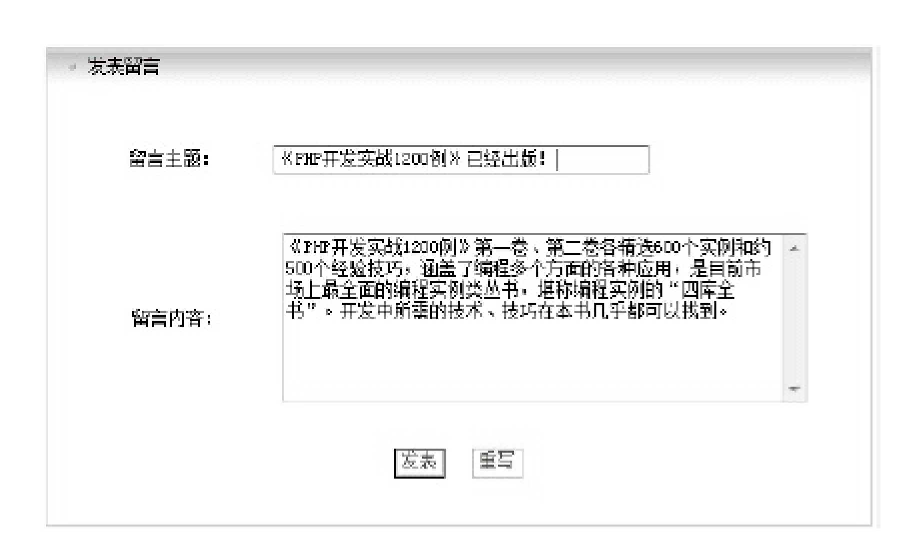


图 14.12 用户添加留言的运行结果

14.7.2 mysql_query()函数执行 SQL 语句

将用户发表的留言保存到 MySQL 数据库是通过 SQL 语言的 insert 语句实现的。PHP 通过 mysql_query()函数向 MySQL 数据库中发送 SQL 命令，所以 mysql_query()函数在数据库编程过程中具有极其重要的作用，下面对该函数进行介绍。

mysql_query()函数用于向与指定的连接标识符所关联的数据库服务器中发送一条查询语句。其语法如下：

```
resource mysql_query (string query [, resource link_identifier])
```

☑ query：必选参数。用于指定向数据库中发送的查询语句。

☑ link_identifier：可选参数。mysql_connect()函数成功连接数据库后所返回的连接标识，如果省略了该参数，则使用最近一次与数据库建立连接后所返回的连接标识。

14.7.3 添加留言模块实现过程

添加留言模块，首先应建立用于留言的表单，该过程实现相对简单，这里不再赘述。

然后，获取表单中提交的数据，对用户输入的留言信息进行敏感词过滤，将用户留言内容保存到数据库中。其具体代码如下：

（代码位置：光盘\TM\Instance\14\saveleaveword.php）

```
<?php
session_start();                                    //启用 session 支持
include_once("conn/conn.php");                      //包含数据库连接文件
include_once("function.php");                       //包含系统功能文件
if(isset($_SESSION["unc"])){                        //对登录用户进行判断
	$sql=mysql_query("select id from tb_user where usernc='".$_SESSION["unc"]."'");
	                                                //查询当前用户数据表中的信息
	$info=mysql_fetch_array($sql);                  //从结果集中获取信息
	$userid=$info['id'];                            //获取用户的 id
}
if(isset($_SESSION['userword'])){
	$sql=mysql_query("select id from tb_adm where userword='".$_SESSION["userword"]."'");
	$info=mysql_fetch_array($sql);
	$userid=$info['id'];
}
if(isset($_POST['content'])){
```

```
❶        if (is_file("filterwords.txt")){                    //判断给定文件名是否为一个正常的文件
         $filter_word = file("filterwords.txt");             //把整个文件读入一个数组中
         $content=$_POST['content'];
         $title=$_POST['title'];
         for($i=0;$i<count($filter_word);$i++){              //应用 for 循环语句对敏感词进行判断
❷                if(preg_match("/".trim($filter_word[$i])."/i",$content)){    //应用正则表达式，判断传递的留言信
                                                                               息中是否含有敏感词
                 echo "<script>alert('留言信息中包含敏感词！');history.back(-1);</script>";
                 exit;
             }
         }
     }
❸        $createtime=date("Y-m-d H:i:s");                    //获取留言时间
         if(mysql_query("insert into tb_leaveword(userid,createtime,title,content)values('$userid','$createtime','$title',
'$content')")){
         echo  "<script>alert('留言发表成功！');window.location.href='index.php?id=".urlencode('查看留言
')."';</script>";
         //在页面中提交的留言信息，写入到数据库留言表中，提示留言成功并跳转到查看留言
     }else{
         echo "<script>alert('留言发表失败！');history.back();</script>";
     }
}
?>
```

说明

❶is_file()函数：判断该函数参数所表示的文件是否存在并且为一个合法文件。

❷ preg_match()函数：preg_match()函数中的“/i”，是指在进行敏感词汇比较时区分字母大小写。

❸date()函数：格式化一个本地时间/日期。

14.8 查看留言模块设计

视频讲解：光盘\TM\Video\第 14 章\查看留言模块设计.exe

14.8.1 查看留言模块概述

用户登录后，单击首页上的“查看留言”链接，将进入浏览页面。在浏览页面中，用户除了可以看到留言信息外，还可以看到对该留言的回复。如果留言或者回复是当前用户所发表，那么还会看到修改的超链接。如果回复是当前用户所发表的，那么除显示上述超链接外，还显示删除的超链接。查看留言页面的运行结果如图 14.13 所示。

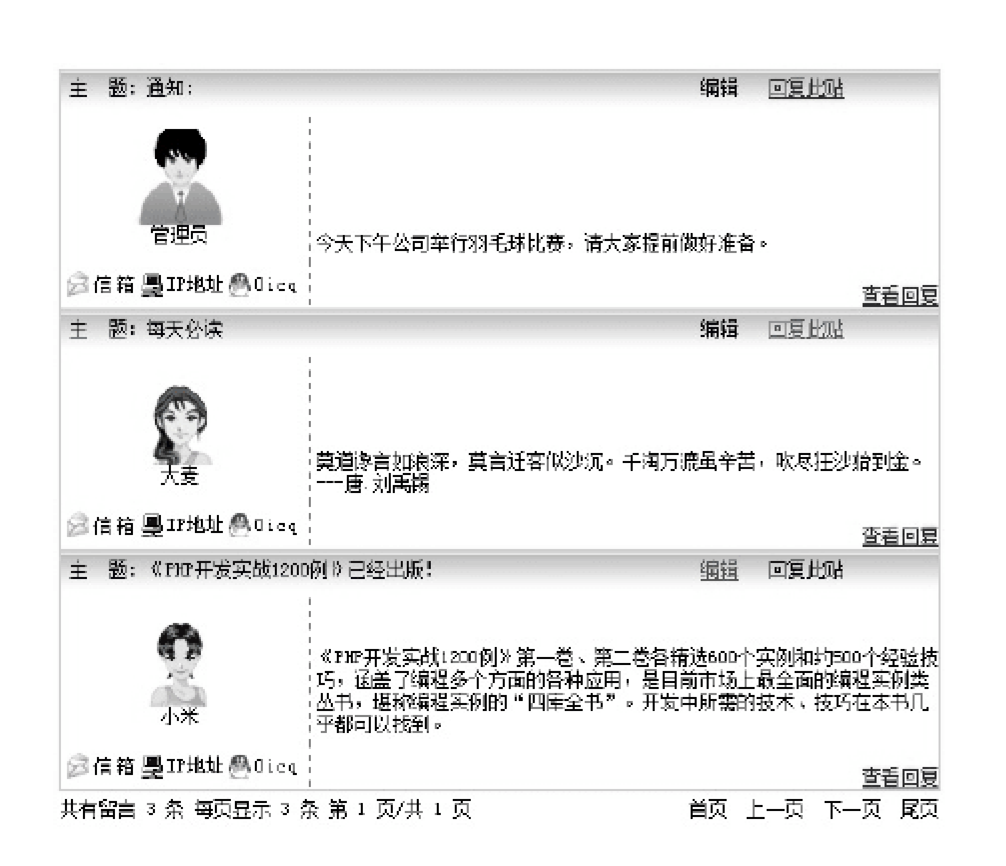


图 14.13 查看留言界面

14.8.2 验证数据类型与取整

用户发表完留言后，可以通过查看留言模块查看用户的所有留言内容。由于用户的留言数目较多，如果在同一页面

中显示所有留言信息则会给用户浏览带来很大的不便，所以通过分页的方式显示用户留言内容是不错的选择。在实现用户留言内容分页显示时，主要应用 is_numeric()函数判断用户通过 GET 方法提交的数据是否为数值型，并通过 ceil()函数对页码数据进行向上取整。

1. is_numeric()函数

如果该函数的参数为数字或数字字符串则返回 True，否则返回 False。语法如下：

```
bool is_numeric ( mixed var )
```

参数 var 为要进行判断的数据。

2. ceil()函数

ceil()函数用于对浮点数进行向上取整。语法如下：

```
float ceil ( float value )
```

参数 value 为要进行向上取整的数据。

14.8.3 查看留言模块实现过程

在查看留言模块中，以分页的形式循环输出留言信息，并且根据对当前用户权限设置一些具体的操作链接是否输出。下面是显示用户留言信息的关键代码。

（代码位置：光盘\TM\Instance\14\lookleaveword.php）

```
<?php
$sql=mysql_query("select count(*) as total from tb_leaveword ",$conn);
        $infos=mysql_fetch_array($sql);
        $total=$infos['total'];                    //获取总留言条数
          if($total==0){                           //如果总留言条数为 0，则给出提示
                echo "<div align=center>对不起，暂无留言！</div>";
                }else{
                 if(!isset($_GET["page"]) || !is_numeric($_GET["page"])){
                     $page=1;                      //判断查询字符串 page 的值是否为空，如果为空则默认显示第 1 页
                   }else{
                     $page=intval($_GET["page"]);
                        }
                 $pagesize=3;                                  //规定每页显示 3 条留言
                 if($total%$pagesize==0){                      //获取总页数
                     $pagecount=intval($total/$pagesize);      //如果获取的总页数是整数，则返回整数值
                 }else{
                     $pagecount=ceil($total/$pagesize);        //获取的总页数不是整数时，进行向上取整
                 }
$sql=mysql_query("select * from tb_leaveword order by createtime desc limit ".($page-1)*$pagesize.",$pagesize
",$conn);
     while($info=mysql_fetch_array($sql)){                     //通过 while 循环显示所有留言
$sql1=mysql_query("select usernc,face,ip,email,qq from tb_user where id='".$info["userid"]."'",$conn);
                                                               //通过传递的 id 值查询当前留言者的个人信息
$info1=mysql_fetch_array($sql1);
?>
                 .......                                       //显示用户信息和用户留言信息
<?php
$adms=mysql_query("select id from tb_adm where userword='".$adm."'",$conn);
                                                               //从管理员数据表中查询登录的版主 id
$re=mysql_fetch_array($adms);
if($re['id']==$info['userid'] && $adm!=""){ //判断当前登录的管理员是不是该条留言的发表者，如果是则显示编辑按钮
?>
          ........                                             //显示编辑按钮
```



```
<?php
    }elseif($user== $info1['usernc'] and $user!=""){        //如果当前用户是该留言的发表者，则显示编辑按钮
?>
        ........                                              //显示编辑按钮
<?php
    }else{
?>
    ........                                                  //编辑按钮为灰色，即不显示
<?php
}
?>
<?php
        if($adm!=""){                                         //管理员登录后，可以对帖子进行回复
?>
      ........                                                //显示回复此帖
<?php
        }else if($user!= $info1['usernc'] and $user!=""){ //当前用户不是该帖子的发表者时，可以对帖子进行回复
?>
      ........                                                //显示回复此帖
<?php
        }else{
?>
<?php
      }
?>
```

上述代码中，实现留言信息的分页显示，是通过 MySQL 数据库的扩展关键字 limit 来实现的，该关键字后跟两个参数。其中，第一个参数用于指定要显示记录的起始位置，而第二个参数用于指定所要显示的记录个数。

在显示用户留言时，根据超链接传递的 page 值来决定所要显示的记录范围。用于实现向页面中传入查询字符串的关键代码如下：

```
<table width="550" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
        <tr>
          <td width="351"><div align="left">共有留言 <?php echo $total;?> 条 每页显示
 <?php echo $pagesize;?> 条 第 <?php echo $page;?> 页/共 <?php echo
$pagecount;?> 页</div></td>

          <td width="199"><div align="right">
            <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=1&id=<?php echo urlencode($id);?>"
class="a1">首页</a> 
            <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=<?php
                if($page>1)              //判断当前页是否大于第一页，如果是则当用户单击“上一页”超
                                          链接时，使变量$page 的值减 1，从而实现向前翻页的功能
                    echo $page-1;
                else
                    echo 1;
            ?>&id=<?php echo urlencode($id);?>" class="a1">上一页</a> 
          <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=<?php
                if($page<$pagecount)     //判断当前页码是否小于总的页数，如果是则当用户单击“下一页”
                                          超链接时，使变量$page 的值加 1，从而实现向前翻页的功能
                  echo $page+1;
                else
                    echo $pagecount;
            ?>&id=<?php echo urlencode($id);?>" class="a1">下一页</a> 
            <a href="<?php  echo $_SERVER["PHP_SELF"]?>?page=<?php echo $pagecount;?>&id=<?php
echo urlencode($id);?>" class="a1">尾页</a></div></td>
        </tr>
```

```
</table>
```

14.9 编辑留言模块设计

14.9.1 编辑留言模块概述

当用户浏览自己发表的留言或回复其他用户留言时，会显示编辑功能。用户可以通过这个编辑链接，对已发表的内容进行修改。编辑留言页面的运行结果如图 14.14 所示。

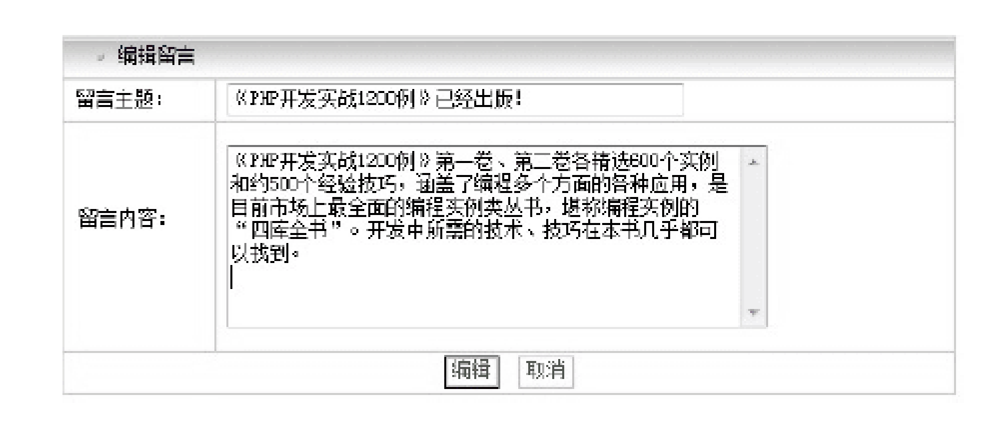


图 14.14 编辑留言页面的运行结果

14.9.2 JavaScript 脚本控制弹出对话框

在本项目中，采用弹出窗口显示预编辑的留言内容。通过 JavaScript 实现弹出窗口，主要是通过调用 window 对象的 open()方法实现，并将实现调用的代码封装到一个单独的自定义方法中，最后通过 HTML 元素的 onclick()事件对该方法进行调用。

open()方法的语法如下：

```
open("url","name","features")
```

该方法参数说明如表 14.6 所示，参数 features 的常规特征如表 14.7 所示。

表 14.6 open()方法参数说明

参数取值	说明
url	可选参数。用于指定弹出新窗口的 URL 地址，如果省略该参数或它的值是一个空字符串，那么新窗口就不显示任何文档
name	新窗口的名字，如果这个名字是一个已经存在的窗口，那么 open()方法就不再创建一个新窗口，而只是返回这个窗口的引用，在这种情况下参数 features 将被忽略
features	该窗口主要是显示窗口的一些特征，该参数如果不设定则显示这个窗口的所有特征。features 的常规特征请参见表 14.7

表 14.7 参数 features 的常规特征

名称	用法
channelmode	指定窗口是否应该以频道的方式显示，取值有 yes/no 或 1/0
fullscreen	指定窗口是否全屏显示，取值有 yes/no 或 1/0
menubar	指定窗口是否有菜单栏，取值有 yes/no 或 1/0
scrollbars	指定窗口是否有水平及竖直滚动条，取值有 yes/no 或 1/0
status	指定窗口是否有状态栏，取值有 yes/no 或 1/0
toolbar	指定窗口是否有工具栏，取值有 yes/no 或 1/0
location	指定窗口是否有地址栏，取值有 yes/no 或 1/0
directions	指定新建窗口是否有标准目录按扭，取值有 yes/no 或 1/0
resizable	指定新窗口大小是否可以调整，取值有 yes/no 或 1/0
top	以像素为单位指定新窗口上沿与屏幕上沿的距离
left	以像素为单位指定新窗口左沿与屏幕左沿的距离



width	以像素为单位指定新窗口的宽度
height	以像素为单位指定新窗口的高度

14.9.3　编辑留言模块实现过程

为了保证用户的留言内容不被他人私自更改，在具体实现该模块功能时，采用每个用户只能对自己的留言内容进行更改的方式。具体实现时，将标识用户身份的 session 变量的值与数据库中该条留言对应的留言者进行比较，如果二者相同，则说明该条留言为当前登录用户所发表的，在该条留言后显示“编辑”按钮。当用户单击“编辑”按钮后，即可实现留言信息的更改，并且在关闭弹出窗口的瞬间，留言信息的显示页面也会自动进行刷新。其代码如下：

（代码位置：光盘\TM\Instance\14\editleaveword.php）

```
<?php
include_once("conn.php");
$id=$_GET["id"];
if($_POST["submit"]!=""){ //判断用户是否单击了“提交”按钮，如果是则执行如下用于实现用户留言信息更改的
                            代码，并刷新父窗口和关闭弹出窗口
  if(MySQL_query("update tb_leaveword set title='".$_POST["title"]."',content='".$_POST["content"]."' where
id='".$_POST["id"]."'",$conn)){
     echo "<script>alert('留言更改成功！');window.opener.location.reload();window.close();</script>";
    }else{
     echo "<script>alert('留言更改失败！');history.back();</script>";
  }
 exit;
}
$sql=MySQL_query("select * from tb_leaveword where id='".$id."'",$conn);
       //如果用户未单击“编辑”按钮，则查询该用户的留言内容，并最终将未编辑的留言内容显示在表单中
$info=MySQL_fetch_array($sql);
?>
```

编辑用户留言模块，用户实现关闭弹出窗口，自动刷新父窗口，其通过如下方法完成。

```
window.opener.location.reload();
```

上述代码的实现原理是在用 window.close()语句关闭弹出窗口前，调用父窗口的 reload()方法实现父窗口的刷新。

14.10　查询留言模块设计

14.10.1　查询留言模块概述

查询留言是对数据库中的数据按条件进行筛选浏览。本留言本提供按主题、内容、留言者 3 种条件，用户可以任选其一作为查询条件进行快速查询。查询留言页面的运行结果如图 14.15 所示。

图 14.15　查询留言页面的运行结果

14.10.2　通过 mysql_fetch_array()函数返回结果集

PHP 中提供了一组操作 MySQL 数据库的函数，应用这些函数可以方便地对 MySQL 数据库进行管理。

mysql_fetch_array()函数的作用是获得数据库中满足 mysql_query()函数中的 SQL 语句的记录，其返回值是一个数组，该数组的下标可以是字段名，也可以是索引下标，数组元素的值是某个字段的内容。该函数会使记录指针自动向下移动，当移动到最后一行将返回一个 False 值。其语法如下：

```
array mysql_fetch_array(int result,int[result_type])
```

☑　result：必选参数。mysql_query()函数向 MySQL 数据库发送查询命令的返回标识。

☑　result_type：可选参数。如果该参数取 MYSQL_BOTH，则该函数将返回一个同时包含关联索引和数字索引的数组；如果该参数取 MYSQL_ASSOC，则返回一个关联索引的数组；如果取 MYSQL_NUM，则返回一个数字索引的数组。

说明　与该函数功能类似的函数还有 MySQL_fetch_rows()、MySQL_result()、MySQL_fetch_object()等，MySQL_fetch_rows()函数返回的数组的下标为数值索引下标。MySQL_result()函数有两个参数，其中第一个参数也是 MySQL_query()的返回结果，而第二个参数可以是字段的偏移量或者字段名，一定注意它返回的结果不是数组，而是 MySQL 结果集中一个单元的内容。MySQL_fetch_object()函数的返回结果是个对象，使用时只能通过字段名来返回结果。

14.10.3　查询留言模块实现过程

当用户按要求添加完查找关键字，并选择了查找方式对表单进行提交后，程序将对用户的提交内容进行判断，并最终显示查询结果。实现该过程的关键代码如下：

（代码位置：光盘\TM\Instance\14\searchword.php）

```
<?php
   if($_POST["submit"]!=""){                    //判断用户是否单击了“查询”按钮，如果是则开始查询工作
      $type=$_POST["type"];
      $keyword=$_POST["keyword"];
      if($type==1){                             //根据用户提交的查询类型，定义查询语句
         $sql=MySQL_query("select * from tb_leaveword where title like '%".$keyword."%'",$conn);
      }elseif($type==2){
         $sql=MySQL_query("select * from tb_leaveword where content like '%".$keyword."%'",$conn);
      }elseif($type==3){
         $sql0=MySQL_query("select id from tb_user where usernc='".$keyword."'",$conn);
         $info0=MySQL_fetch_array($sql0);
         $sql=MySQL_query("select * from tb_leaveword where userid='".$info0["id"]."'",$conn);
```

```
        }
        $info=MySQL_fetch_array($sql);
        if($info==false){                              //判断是否查找到相关内容，如果没查找到则显示提示信息
          echo "<br><br><div align=center>对不起，没有查找到您要查找的内容！</div>";
        }else{
            do{                                        //通过循环显示查询结果
?>
        ……//显示查找内容
 <?php
            }while($info=MySQL_fetch_array($sql));
         }
        }
 ?>
```

14.11　版主模块设计

视频讲解：光盘\TM\Video\第 14 章\版主模块设计.exe

14.11.1　版主模块概述

为了更好地管理和维护留言本，针对留言本设置了一个管理员，该管理员不在后台进行操作，而是在前台为管理员设置特殊的权限，也可以称之为版主。其实现的原理是：需要在 MySQL 数据库中，手动添加一条系统版主的数据，版主想要对留言本进行管理时，只需单击首页上版主管理超链接，登录后即可。版主登录界面如图 14.16 所示，版主浏览如图 14.17 所示。

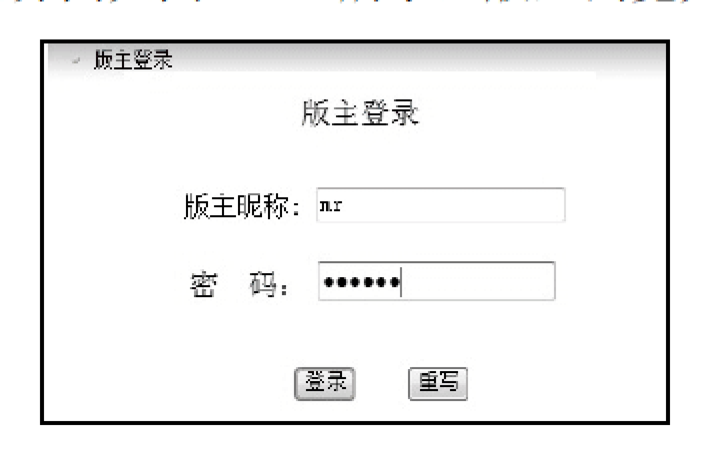


图 14.16　版主登录界面



图 14.17　版主浏览界面

14.11.2　验证登录用户是否是版主

版主单击首页上的版主管理进入版主登录界面。在版主登录模块中，首先判断版主名称和密码是否为空，然后获取表单提交的值，与数据库存储的值进行比较，判断提交的版主名称与密码是否与数据库中的版主名称与密码相对应，如果是则登录成功，反之则登录失败。

版主成功登录后，可以对用户提交的留言进行管理操作，例如，全主题删除、删除回复留言、对用户的留言进行回复等。所谓全主题删除，主要是删除留言及该留言的回复信息。

其操作原理是：当版主单击删除按钮后，将弹出一个对话框提示版主是否真正删除该条留言及回复，如果用户单击对话框中的“确定”按钮，则该条留言及回复将被删除，反之不做任何操作。其主要代码如下：

```
<?php
      $sql="select *   from tb_user where id='".$rows['userid']."'";
      $res=mysql_query($sql,$conn);
```

```
        $rew=mysql_fetch_array($res);
        echo unhtml($rew['usernc']);
?>
                    //表单元素
<?php
        $sqls="select *   from tb_replyword where leave_id='".$rows['id']."'";
        $rs=mysql_query($sqls,$conn);
        $rw=mysql_num_rows($rs);
        echo $rw;
?>
<a href="javascript:if(window.confirm('确定删除该留言信息么？')==true){window.location.href='deleteleaveword.php?
del_id=<?php echo $rows['id']; ?>';}">删除
……//表单元素
```

14.11.3　版主管理模块实现过程

为了保持留言本的正常运行，防止操作权限的扩大，在版主登录处理页中将使用 JavaScript 脚本限制用户与管理员在同一台计算机上登录的情况。版主登录处理页的具体代码如下：

（代码位置：光盘\TM\Instance\14\chklogin.php）

```
<?php
session_start();
include("conn/conn.php");
if(isset($_SESSION['unc'])){
        echo "<script>alert('在同一台机器上，不允许同时使用用户名和管理员进行登录！'); window.location.href=
'index.php';</script>";
}else{
        if(isset($_POST['Submit']) && $_POST['Submit']=="登录") {
                if($_POST['username']!="" && $_POST['password']!=""){
                        $check="select  userword  from  tb_adm  where  userword='".$_POST['username']."'and
password='".$_POST['password']."'";
                        $result=mysql_query($check,$conn);
                        $info=mysql_num_rows($result);
                        if($info==1){
                                $_SESSION["userword"]=$_POST['username'];
                                echo "<script>alert('登录成功'); window.location.href='admin_browse.php';</script>";
                        }else{
                                echo "<script>alert('登录失败，不是版主账号'); window.location.href='index.php';</script>";

                        }
                }
        }
}
?>
```

当版主确认删除全主题留言后，将进入全主题删除处理页面对全主题留言进行删除。实现全主题删除的代码如下：

（代码位置：光盘\TM\Instance\14\guestbook\deleteleaveword.php）

```
<?php
session_start();                                //启用 session 支持
include_once("conn/conn.php");                  //包含数据库链接文件
if(isset($_SESSION['userword']) && isset($_GET["del_id"])){     //对用户进行判断，是管理员则执行全主题删除
        $result=mysql_query("delete from tb_leaveword where id='".$_GET["del_id"]."'",$conn);
                                                //通过 id 传递的值，删除留言表中的留言
```

```
        $res=mysql_query("select * from tb_replyword where leave_id='".$_GET["del_id"]."'",$conn);
                                                        //查询回复表中对该主题的回复
        if(mysql_num_rows($res)>0){           //当回复表中的记录数大于 0 时，查询回复表中对该主题的回复
                $results=mysql_query("delete from tb_replyword where leave_id='".$_GET["del_id"]."'",$conn);
                if($result==true and $results==true){                        //当主题和回复都存在的时，执行全主题删除
                        echo "<script>alert('留言删除成功！');history.back();</script>";
                }else{
                        echo "<script>alert('留言删除失败！');history.back();</script>";
                }
        }else{
                if($result){                            //删除发布留言
                        echo "<script>alert('留言删除成功！');history.back();</script>";
                }else{
                        echo "<script>alert('留言删除失败！');history.back();</script>";
                }
        }
}else{
        echo "<script>alert('您没有删除权限！');history.back();</script>";
}
?>
```

注意 笔者为本书留言本设置的版主名称为 MR，密码是 mrsoft。

14.12　小　　结

本章主要对留言本的设计思路和业务流程进行介绍，并详细介绍了 PHP 连接和管理 MySQL 数据库的方法及留言本的各模块的实现过程。通过本章的学习，不仅可以提高读者自身的基础，而且可以全面掌握适合于中小型企业应用的留言本的制作方法，最终可以在此基础上进行延伸，制作出更有规模的 Web 项目。



中级开发

第15章

MySQL 存储过程和函数

（视频讲解：32 分钟）

存储过程和函数是在数据库中定义一些SQL语句的集合，然后直接调用这些存储过程和函数来执行已经定义好的SQL语句。存储过程和函数可以避免开发人员重复编写相同的SQL语句。而且，存储过程和函数是在MySQL服务器中存储和执行的。可以减少客户端和服务器端的数据传输。本章将介绍存储过程和函数的含义、作用，还可以了解创建、使用、查看、修改及删除存储过程及函数的方法。

通过阅读本章内容，你可以：

- 了解流程控制语句的使用
- 了解 MySQL 存储过程和函数中光标的使用和一般步骤
- 掌握 MySQL 中存储过程和存储函数的创建
- 掌握 MySQL 存储过程应用函数的参数使用方法
- 掌握存储过程和函数的调用、查看、修改和删除

15.1　创建存储过程和存储函数

在数据库系统中，为了保证数据的完整性、一致性，同时也为提高其应用性能，大多数据库常采用存储过程和存储函数技术。MySQL 在 5.0 版本后，也应用了存储过程和存储函数，存储过程和存储函数经常是一组 SQL 语句的组合，这些语句被当作整体存入 MySQL 数据库服务器中。用户定义的存储函数不能用于修改全局库状态，但该函数可从查询中被唤醒调用，也可以像存储过程一样通过语句执行。随着 MySQL 技术的日趋完善，存储过程将和存储函数在以后的项目中被到广泛应用。

15.1.1　创建存储过程

视频讲解：光盘\TM\Video\第 15 章\视频 15.1\创建存储过程.exe

在 MySQL 中，创建存储过程的基本形式如下：

```
CREATE PROCEDURE sp_name ([proc_parameter[,...]])
   [characteristic ...] routine_body
```

其中，sp_name 参数是存储过程的名称；proc_parameter 表示存储过程的参数列表；characteristic 参数指定存储过程的特性；routine_body 参数是 SQL 代码的内容，可以用 begin…end 来标识 SQL 代码的开始和结束。

技巧 proc_parameter 中的参数由 3 部分组成，它们分别是输入输出类型、参数名称和参数类型。其形式为[IN | OUT | INOUT]param_name type。其中，IN 表示输入参数；OUT 表示输出参数；INOUT 表示既可以输入也可以输出；param_name 参数是存储过程参数名称；type 参数指定存储过程的参数类型，该类型可以为 MySQL 数据库的任意数据类型。

一个存储过程包括名字、参数列表，还可以包括很多 SQL 语句集。下面创建一个存储过程，其代码如下：

```
delimiter
create procedure proc_name (in parameter integer)
begin
declare variable varchar(20);
if parameter=1 then
set variable='MySQL';
else
set variable=’PHP’;
end if;
insert into tb (name) values (variable);
end;
```

MySQL 中存储过程的建立以 create procedure 关键字开始，后面仅跟存储过程的名称和参数。MySQL 的存储过程名称不区分大小写，如 PROCE1()和 proce1()代表同一存储过程名。存储过程名或存储函数名不能与 MySQL 数据库中的内建函数重名。

MySQL 存储过程的语句块以 begin 开始，以 end 结束。语句体中可以包含变量的声明、控制语句、SQL 查询语句等。由于存储过程内部语句要以分号结束，所以在定义存储过程前，应将语句结束标志“;”更改为其他字符，并且应降低该字符在存储过程中出现的几率，更改结束标志可以用 delimiter 关键字定义。例如：

```
mysql>delimiter //
```

存储过程创建后，可用如下语句进行删除，参数 proc_name 指存储过程名。

```
drop procedure proc_name
```

下面创建一个名称为 count_of_student 的存储过程。首先，创建一个名称为 students 的 MySQL 数据库，然后创建一个名为 studentinfo 的数据表。数据表结构如表 15.1 所示。

表 15.1　studentinfo 数据表结构

字　段　名	类型（长度）	默　　认	额　　外	说　　明
sid	INT(11)		auto_increment	主键自增型 sid
name	VARCHAR(50)			学生姓名
age	VARCHAR(11)			学生年龄
sex	VARCHAR(2)			学生性别
tel	BIGINT(11)			联系电话

例 15.1　创建一个名称为 count_of_student 的存储过程，统计 studentinfo 数据表中的记录数。**（实例位置：光盘\TM\Instance\15\15.1）**

代码如下：

```
delimiter //
create procedure count_of_student(OUT count_num INT)
reads sql data
begin
select count(*) into count_num from studentinfo;
end
//
```

在上述代码中，定义一个输出变量 zzELECT 语句从 studentinfo 表中获取记录总数，最后将结果传递给变量 count_num。存储过程的执行结果如图 15.1 所示。

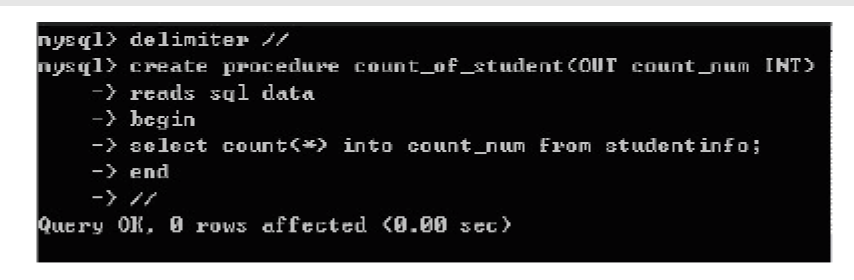


图 15.1　存储过程的执行结果

代码执行完毕后，没有报出任何出错信息就表示存储函数已经创建成功。以后就可以调用这个存储过程，数据库中会执行存储过程中的 SQL 语句。

说明　MySQL 中默认的语句结束符为分号，存储过程中的 SQL 语句需要分号来结束。为了避免冲突，首先用“DELIMITER //”将 MySQL 的结束符设置为//。最后再用“DELIMITER;”来将结束符恢复成分号。这与创建触发器时是一样的。

15.1.2　创建存储函数

视频讲解：光盘\TM\Video\第 15 章\创建存储函数.exe

创建存储函数与创建存储过程大体相同。其创建存储函数的基本形式如下：

```
CREATE FUNCTION sp_name ([func_parameter[,...]])
    RETURNS type
   [characteristic ...] routine_body
```

创建存储函数的参数说明如表 15.2 所示。

表 15.2　创建存储函数的参数说明

参　　数	说　　明
sp_name	存储函数的名称
fun_parameter	存储函数的参数列表



续表

参　　数	说　　明
RETURNS type	指定返回值的类型
characteristic	指定存储过程的特性
routine_body	SQL 代码的内容

func_parameter 可以由多个参数组成，其中每个参数均由参数名称和参数类型组成。其结构如下：

```
param_name type
```

其中，param_name 参数是存储函数的函数名称；type 参数用于指定存储函数的参数类型。该类型可以是 MySQL 数据库所支持的类型。

例 15.2 同样，应用 studentinfo 表，创建名为 name_of_student 的存储函数。**（实例位置：光盘\TM\Instance\15\15.2）**

其代码如下：

```
delimiter //
create function name_of_student(std_id INT)
returns varchar(50)
begin
return(select name from studentinfo where sid=std_id);
end
//
```

上述代码中，存储函数的名称为 name_of_student，该函数的参数为 std_id，返回值是 VARCHAR 类型。该函数实现从 studentinfo 表查询与 std_id 相同 sid 值的记录，并将学生名称字段 name 中的值返回。存储函数的执行结果如图 15.2 所示。

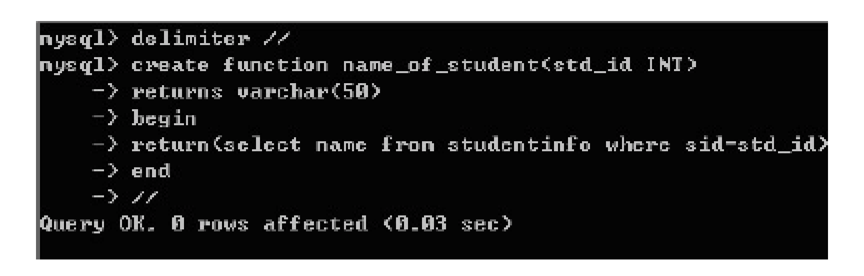


图 15.2　创建 name_of_student()存储函数

15.1.3　变量的应用

视频讲解：光盘\TM\Video\第 15 章\变量的应用.exe

MySQL 存储过程中的参数主要有局部参数和会话参数两种，这两种参数又可以被称为局部变量和会话变量。局部变量只在定义该局部变量的 begin…end 范围内有效，会话变量在整个存储过程范围内均有效。

1．局部变量

局部变量以 declare 关键字声明，后跟变量名和变量类型，例如：

```
declare a int
```

当然在声名局部变量时也可以用 default 关键字为变量指定默认值，例如：

```
declare a int default 10
```

下面的代码为读者展示如何在 MySQL 存储过程中定义局部变量以及其使用方法。在该例中，分别在内层和外层 begin…end 块中都定义同名的变量 x，按照语句从上到下执行的顺序，如果变量 x 在整个程序中都有效，则最终结果应该都为 inner，但真正的输出结果却不同，这说明在内部 begin…end 块中定义的变量只在该块内有效。

例 15.3 该例说明局部变量只在某个 begin…end 块内有效。**（实例位置：光盘\TM\Instance\15\15.3）**

代码如下：

```
delimiter //
create procedure p1()
begin
declare x char(10) default 'outer ';
begin
```

```
declare x char(10) default 'inner ';
select x;
end;
select x;
end;
//
```

上述代码的运行结果如图 15.3 所示。

应用 MySQL 调用该存储过程的运行结果如图 15.4 所示。

```
mysql> delimiter //
mysql> create procedure p1()
    -> begin
    -> declare x char(10) default 'outer ';
    -> begin
    -> declare x char(10) default 'inner ';
    -> select x;
    -> end;
    -> select x;
    -> end;
    -> //
Query OK, 0 rows affected (0.01 sec)
```

图 15.3　定义局部变量的运行结果

```
mysql> call p1()//
+-------+
| x     |
+-------+
| inner |
+-------+
1 row in set (0.01 sec)

+-------+
| x     |
+-------+
| outer |
+-------+
1 row in set (0.03 sec)

Query OK, 0 rows affected (0.05 sec)
```

图 15.4　调用存储过程 pl()的运行结果

2．全局变量

MySQL 中的会话变量不必声明即可使用，会话变量在整个过程中有效，会话变量名以字符“@”作为起始字符。下面的代码为会话变量的使用方法。

例 15.4　在该例中，分别在内部和外部 begin…end 块中都定义了同名的会话变量@t，并且最终输出结果相同，从而说明会话变量的作用范围为整个程序。**（实例位置：光盘\TM\Instance\15\15.4）**

设置全局变量的代码如下：

```
delimiter //
create procedure p2()
begin
set @t=1;
begin
set @t=2;
select @t;
end;
select @t;
end;
//
```

上述代码的运行结果如图 15.5 所示。

应用 MySQL 调用该存储过程的运行结果如图 15.6 所示。

```
mysql> delimiter //
mysql> create procedure p2()
    -> begin
    -> set @t=1;
    -> begin
    -> set @t=2;
    -> select @t;
    -> end;
    -> select @t;
    -> end;
    -> //
Query OK, 0 rows affected (0.00 sec)
```

图 15.5　设置全局变量



图 15.6　调用存储过程 p2()运行结果

3．为变量赋值

MySQL 中可以使用 DECLARE 关键字来定义变量。定义变量的基本语法如下：



```
DECLARE var_name[,…] type [DEFAULT value]
```

其中，DECLARE 是用来声明变量的；var_name 参数是设置变量的名称。如果用户需要，也可以同时定义多个变量。type 参数用来指定变量的类型；DEFAULT value 的作用是指定变量的默认值，不对该参数进行设置时，其默认值为 NULL。

MySQL 中可以使用 SET 关键字为变量赋值。SET 语句的基本语法如下：

```
SET var_name=expr[,var_name=expr]…
```

其中，SET 关键字是用来为变量赋值；var_name 参数是变量的名称；expr 参数是赋值表达式。一个 SET 语句可以同时为多个变量赋值，各个变量的赋值语句之间用“,”隔开。例如，为变量 mr_soft 赋值，代码如下：

```
SET mr_soft=10;
```

另外 MySQL 中还可以应用另一种方式为变量赋值。其语法结构如下：

```
SELECT col_name[,…] INTO var_name[,…] FROM table_name where condition
```

其中，col_name 参数标识查询的字段名称；var_name 参数是变量的名称；table_name 参数为指定数据表的名称；condition 参数为指定查询条件。例如，从 studentinfo 表中查询 name 为 LeonSK 的记录。将该记录下的 tel 字段内容赋值给变量 customer_tel。其关键代码如下：

```
SELECT tel INTO customer_tel FROM studentinfo WHERE name= 'LeonSK ';
```

说明 上述赋值语句必须存在于创建的存储过程中。且需将赋值语句放置在 BEGIN…END 之间。若脱离此范围，该变量将不能使用或被赋值。

15.1.4　光标的运用

视频讲解：光盘\TM\Video\第 15 章\光标的运用.exe

通过 MySQL 查询数据库，其结果可能为多条记录。在存储过程和函数中使用光标可以实现逐条读取结果集中的记录。光标使用包括声明光标（DECLARE CURSOR）、打开光标（OPEN CURSOR）、使用光标（FETCH CURSOR）和关闭光标（CLOSE CURSIR）。值得一提的是，光标必须声明在处理程序之前，且声明在变量和条件之后。

1．声明光标

在 MySQL 中，声明光标仍使用 DECLARE 关键字。其语法如下：

```
DECLARE cursor_name CURSOR FOR select_statement
```

其中，cursor_name 是光标的名称，光标名称使用与表名同样的规则；select_statement 是一个 SELECT 语句，返回一行或多行数据。其中这个语句也可以在存储过程中定义多个光标，但是必须保证每个光标名称的唯一性，即每一个光标必须有自己唯一的名称。

通过上述定义来声明光标 info_of_student，其代码如下：

```
DECLARE info_of_student CURSOR FOR SELECT
sid,name,age,sex,age
FROM studentinfo
WHERE sid=1;
```

技巧 这里 SELECT 子句中不能包含 INTO 子句，并且光标只能在存储过程或存储函数中使用。上述代码并不能单独执行。

2．打开光标

在声明光标之后，要从光标中提取数据，必须首先打开光标。在 MySQL 中，使用 OPEN 关键字来打开

光标。其基本的语法如下：

```
OPEN cursor_name
```

其中，cursor_name 参数表示光标的名称。在程序中，一个光标可以打开多次。由于可能在用户打开光标后，其他用户或程序正在更新数据表。所以可能会导致用户在每次打开光标后，显示的结果都不同。

打开上面已经声明的光标 info_of_student，其代码如下：

```
OPEN info_of_student
```

3．使用光标

光标在顺利打开后，可以使用 FETCH…INTO 语句来读取数据。其语法如下：

```
FETCH   cursor_name INTO var_name[,var_name]…
```

其中，cursor_name 代表已经打开的光标名称；var_name 参数表示将光标中的 SELECT 语句查询出来的信息存入该参数中。var_name 是存放数据的变量名，必须在声明光标前定义好。FETCH…INTO 语句与 SELECT…INTO 语句具有相同的意义。

将已打开的光标 info_of_student 中 SELECT 语句查询出来的信息存入 tmp_name 和 tmp_tel 中。其中 tmp_name 和 tmp_tel 必须在使用前定义。其代码如下：

```
FETCH info_of_student INTO tmp_name,tmp_tel;
```

4．关闭光标

光标使用完毕后，要及时关闭，在 MySQL 中采用 CLOSE 关键字关闭光标。其语法格式如下：

```
CLOSE cursor_name
```

cursor_name 参数表示光标名称。

下面关闭已打开的光标 info_of_student。代码如下：

```
CLOSE info_of_student
```

技巧 对于已关闭的光标，在其关闭之后则不能使用 FETCH 来使用光标。光标在使用完毕后一定要关闭。

15.2　流程控制语句

在 MySQL 中，常见的过程式 SQL 语句可以用在一个存储过程体中。其中包括 IF 语句、CASE 语句、LOOP 语句、WHILE 语句、ITERATE 语句和 LEAVE 语句，它们可以进行流程控制。

15.2.1　IF 语句

IF 语句用来进行条件判断，根据不同的条件执行不同的操作。该语句在执行时首先判断 IF 后的条件是否为真，如果为真则执行 THEN 后的语句，如果为假则继续判断 IF 语句直到为真为止，当以上都不满足时则执行 ELSE 语句后的内容。IF 语句表示形式如下：

```
IF condition THEN
          …
[ELSE condition THEN]
  …
[ELSE]
…
ENDIF
```



例 15.5 下面通过 if…then…else 结构首先判断传入参数的值是否为 1，如果是则输出 1，如果不是则再判断该传入参数的值是否为 2，如果是则输出 2，当以上条件都不满足时输出 3。**（实例位置：光盘\TM\Instance\15\15.5）**

其代码如下：

```
delimiter //
create procedure example_if(in x int)
begin
if x=1 then
select 1;
elseif x=2 then
select 2;
else
select 3;
end if;
end
//
```

以上代码的运行结果如图 15.7 所示。

通过 MySQL 调用该存储过程，其运行结果如图 15.8 所示。

```
mysql> delimiter //
mysql> create procedure example_if(in x int)
    -> begin
    -> if x=1 then
    -> select 1;
    -> elseif x=2 then
    -> select 2;
    -> else
    -> select 3;
    -> end if;
    -> end
    -> //
Query OK, 0 rows affected (0.00 sec)
```

图 15.7　应用 IF 语句的存储过程

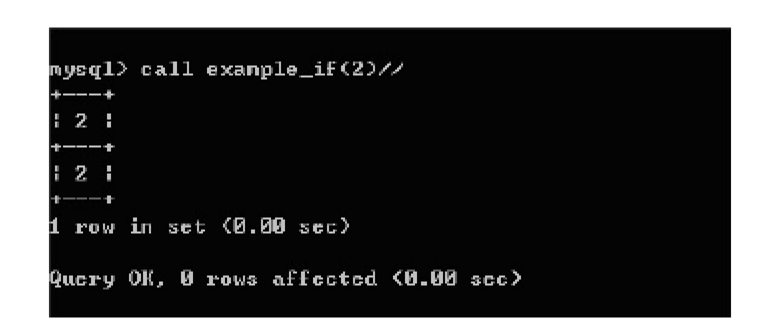


图 15.8　调用 example_if()存储过程

15.2.2　CASE 语句

CASE 语句为多分支语句结构，该语句首先从 when 后的 value 中查找与 case 后的 value 相等的值，如果查找到则执行该分支的内容，否则执行 else 后的内容。CASE 语句表示形式如下：

```
CASE value
        WHEN value THEN…
        [WHEN valueTHEN…]
        [ELSE…]
END CASE
```

其中，value 参数表示条件判断的变量；WHEN…THEN 中的 value 参数表示变量的取值。

CASE 语句还有另一种语法表示结构：

```
CASE
        WHEN value THEN…
        [WHEN valueTHEN…]
        [ELSE…]
END CASE
```

技巧 一个 CASE 语句经常可以充当一个 IF…THEN…ELSE 语句。

例 15.6 下面通过 CASE 语句首先判断传入参数的值是否为 1，如果条件成立则输出 1，如果条件不成

立则再判断该传入参数的值是否为 2，如果成立则输出 2，当以上条件都不满足时输出 3。**（实例位置：光盘\TM\Instance\15\15.6）**

代码如下：

```
delimiter //
create procedure example_case(in x int)
begin
case x
when 1 then select 1;
when 2 then select 2;
else select 3;
end case;
end
//
```

运行该实例的结果如图 15.9 所示。

调用该存储过程，其运行结果如图 15.10 所示。

```
mysql> delimiter //
mysql> create procedure example_case(in x int)
    -> begin
    -> case x
    -> when 1 then select 1;
    -> when 2 then select 2;
    -> else select 3;
    -> end case;
    -> end
    -> //
Query OK, 0 rows affected (0.00 sec)
```

图 15.9　应用 CASE 语句的存储过程

```
mysql> call example_case(3)//
+---+
| 3 |
+---+
| 3 |
+---+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

图 15.10　调用 example_case()存储过程

15.2.3　WHILE 循环语句

WHILE 循环语句执行时首先判断 condition 条件是否为真，如果是则执行循环体，否则退出循环。该语句表示形式如下：

```
while condition do
…
end while;
```

例 15.7　下面应用 WHILE 语句求前 100 项的和。首先定义变量 i 和 s，分别用来控制循环的次数和保存前 100 项的和，当变量 i 的值小于或等于 100 时，使 s 的值加 i，并同时使 i 的值增 1。直到 i 大于 100 时退出循环并输出结果。**（实例位置：光盘\TM\Instance\15\15.7）**

代码如下：

```
delimiter //
create procedure example_while (out sum int)
begin
declare i int default 1;
declare s int default 0;
while i<=100 do
set s=s+i;
set i=i+1;
end while;
set sum=s;
end
//
```

运行以上代码的结果如图 15.11 所示。

调用该存储过程，调用语句如下：

```
call example_while(@s)
mysql>select @s
```

调用该存储过程的结果如图 15.12 所示。

```
mysql> delimiter //
mysql> create procedure example_while (out sum int)
    -> begin
    -> declare i int default 1;
    -> declare s int default 0;
    -> while i<=100 do
    -> set s=s+i;
    -> set i=i+1;
    -> end while;
    -> set sum=s;
    -> end
    -> //
Query OK, 0 rows affected (0.00 sec)
```

图 15.11　应用 WHILE 语句的存储过程

```
mysql> select @s//
+------+
| @s   |
+------+
| 5050 |
+------+
1 row in set (0.00 sec)

mysql>
```

图 15.12　调用 example_while()存储过程

15.2.4　LOOP 循环语句

该循环没有内置的循环条件，但可以通过 LEAVE 语句退出循环。LOOP 语句表示形式：

```
loop
…
end loop
```

LOOP 允许某特定语句或语句群的重复执行，实现一个简单的循环构造，其中中间省略的部分是需要重复执行的语句。在循环内的语句一直重复直至循环被退出，退出循环应用 LEAVE 语句。

LEAVE 语句经常和 BEGIN…END 或循环一起使用，其结构如下：

```
LEAVE label
```

label 是语句中标注的名字，这个名字是自定义的。加上 LEAVE 关键字就可以用来退出被标注的循环语句。

例 15.8　下面应用 LOOP 语句求前 100 项的和。首先定义变量 i 和 s，分别用来控制循环的次数和保存前 100 项的和，进入该循环体后首先使 s 的值加 i，之后使 i 加 1 并进入下次循环，直到 i 大于 100，通过 leave 语句退出循环并输出结果。**（实例位置：光盘\TM\Instance\15\15.8）**

代码如下：

```
delimiter //
create procedure example_loop (out sum int)
begin
declare i int default 1;
declare s int default 0;
loop_label:loop
set s=s+i;
set i=i+1;
if i>100 then
leave loop_label;
end if;
end loop;
set sum=s;
end
//
```

上述代码的运行结果如图 15.13 所示。

调用名称为 example_loop 的存储过程，其代码如下：

```
call example_loop(@s)
select @s
```

运行结果如图 15.14 所示。

```
mysql> delimiter //
mysql> create procedure example_loop (out sum int)
    -> begin
    -> declare i int default 1;
    -> declare s int default 0;
    -> loop_label:loop
    -> set s=s+i;
    -> set i=i+1;
    -> if i>100 then
    -> leave loop_label;
    -> end if;
    -> end loop;
    -> set sum=s;
    -> end
    -> //
Query OK, 0 rows affected (0.00 sec)
```

图 15.13　应用 LOOP 语句创建存储过程

```
mysql> call example_loop(@s)//
Query OK, 0 rows affected (0.00 sec)

mysql> select @s//
+------+
| @s   |
+------+
| 5050 |
+------+
1 row in set (0.00 sec)
```

图 15.14　调用 example_loop()存储过程

15.2.5　REPEAT 循环语句

该语句先执行一次循环体，之后判断 condition 条件是否为真，若为真则退出循环，否则继续执行循环。repeat 语句表示形式如下：

```
REPEAT
  …
UNTIL condition
END REPEAT
```

例 15.9　下面应用 repeat 语句求前 100 项和。首先定义变量 i 和 s，分别用来控制循环的次数和保存前 100 项的和，进入循环体后首先使 s 的值加 i，之后使 i 的值加 1，直到 i 大于 100 时退出循环并输出结果。**（实例位置：光盘\TM\Instance\15\15.9）**

代码如下：

```
delimiter //
create procedure example_repeat (out sum int)
begin
declare i int default 1;
declare s int default 0;
repeat
set s=s+i;
set i=i+1;
until i>100
end repeat;
set sum=s;
end
//
```

以上代码的运行结果如图 15.15 所示。

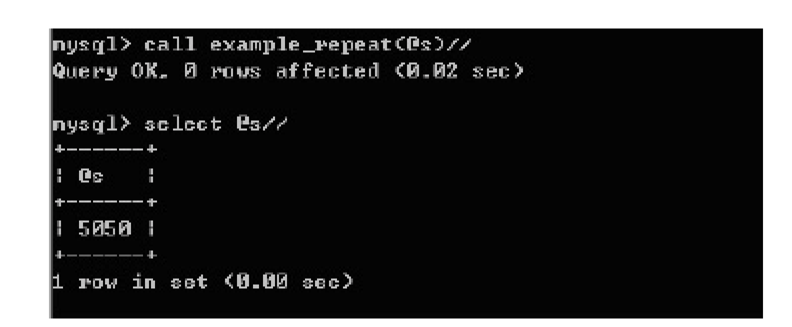
调用该存储过程，相关代码如下：

```
call example_repeat(@s)
select @s
```

调用该存储过程的运行结果如图 15.16 所示。

```
mysql> delimiter //
mysql> create procedure example_repeat (out sum int)
    -> begin
    -> declare i int default 1;
    -> declare s int default 0;
    -> repeat
    -> set s=s+i;
    -> set i=i+1;
    -> until i>100
    -> end repeat;
    -> set sum=s;
    -> end
    -> //
Query OK, 0 rows affected (0.00 sec)
```

图 15.15　应用 REPEAT 语句创建存储过程



```
mysql> call example_repeat(@s)//
Query OK, 0 rows affected (0.02 sec)

mysql> select @s//
+------+
| @s   |
+------+
| 5050 |
+------+
1 row in set (0.00 sec)
```

图 15.16　调用 example_repeat()存储过程

循环语句中还有一个 ITERATE 语句，它可以出现在 LOOP、REPEAT 和 WHILE 语句内，其意为再次循环。该语句格式如下：

```
ITERATE label
```

该语句的格式与 LEAVE 大同小异，区别在于 LEAVE 语句是离开一个循环，而 ITERATE 语句是重新开始一个循环。

注意　与一般程序设计流程控制不同的是存储过程并不支持 FOR 循环。

15.3　调用存储过程和存储函数

视频讲解：光盘\TM\Video\第 15 章\调用存储过程和存储函数.exe

存储过程和存储函数都是存储在服务器的 SQL 语句的集合。要使用这些已经定义好的存储过程和存储函数就必须要通过调用的方式来实现。

15.3.1　调用存储过程

存储过程的调用在前面的实例中多次被用到。MySQL 中使用 CALL 语句来调用存储过程。调用存储过程后，数据库系统将执行存储过程中的语句，然后将结果返回给输出值。CALL 语句的基本语法形式如下：

```
CALL sp_name([parameter[,…]]);
```

其中，sp_name 是存储过程的名称；parameter 是存储过程的参数。

15.3.2　调用存储函数

在 MySQL 中，存储函数的使用方法与 MySQL 内部函数的使用方法基本相同。用户自定义的存储函数与 MySQL 内部函数性质相同。区别在于存储函数是用户自定义的，而内部函数由 MySQL 自带。其语法结构如下：

```
SELECT function_name([parameter[,…]]);
```

15.4　查看存储过程和函数

视频讲解：光盘\TM\Video\第 15 章\查看存储过程和函数.exe

存储过程和函数创建以后，用户可以查看存储过程和函数的状态和定义。用户可以通过 SHOW STATUS 语句查看存储过程和函数状态，也可以通过 SHOW CREATE 语句来查看存储过程和函数的定义。

15.4.1　SHOW STATUS 语句

在 MySQL 中可以通过 SHOW STATUS 语句查看存储过程和函数的状态。其基本语法结构如下：

```
SHOW {PROCEDURE | FUNCTION}STATUS[LIKE 'pattern']
```

其中，PROCEDURE 参数表示查询存储过程；FUNCTION 参数表示查询存储函数；LIKE 'pattern'参数用来匹配存储过程或函数名称。

15.4.2 SHOW CREATE 语句

MySQL 中可以通过 SHOW CREATE 语句来查看存储过程和函数的状态。其语法结果如下：

```
SHOW CREATE{PROCEDURE | FUNCTION } sp_name;
```

其中，PROCEDURE 参数表示存储过程；FUNCTION 参数表示查询存储函数；sp_name 参数表示存储过程或函数的名称。

例 15.10 下面查询名为 count_of_student 的存储过程。**（实例位置：光盘\TM\Instance\15\15.10）**

其代码如下：

```
show create procedure count_of_student ;
```

其运行结果如图 15.17 所示。

```
1 row in set (0.02 sec)

mysql> show create procedure  count_of_student//
+------------------+---------------------+---------------------------------------
-------------------------------------------------------------------------------
-----------------------------------------+
| Procedure        | sql_mode            | Create Procedure

                                         |
+------------------+---------------------+---------------------------------------
-------------------------------------------------------------------------------
-----------------------------------------+
| count_of_student | NO_AUTO_CREATE_USER | CREATE DEFINER=`root`@`localhost` PRO
CEDURE `count_of_student`(OUT count_num INT)
    READS SQL DATA
begin
select count(*) into count_num from studentinfo;
end ;
```

图 15.17 应用 SHOW CREATE 语句查看存储过程

查询结果显示存储过程的定义、字符集等信息。

技巧 SHOW STATUS 语句只能查看存储过程或函数所操作的数据库对象，如存储过程或函数的名称、类型、定义者、修改时间等信息，并不能查询存储过程或函数的具体定义。如果需要查看详细定义，需要使用 SHOW CREATE 语句。

15.5 修改存储过程和函数

视频讲解：光盘\TM\Video\第 15 章\修改存储过程和函数.exe

修改存储过程和存储函数是指修改已经定义好的存储过程和函数。MySQL 中通过 ALTER PROCEDURE 语句来修改存储过程，通过 ALTER FUNCTION 语句来修改存储函数。

MySQL 中修改存储过程和函数的语法形式如下：

```
ALTER {PROCEDURE | FUNCTION} sp_name [characteristic ...]
characteristic:
{ CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
  | SQL SECURITY { DEFINER | INVOKER }
  | COMMENT 'string'
```

其参数说明如表 15.3 所示。

表 15.3　修改存储过程和函数的语法的参数说明

参　数	说　明
sp_name	存储过程或函数的名称
characteristic	指定存储函数的特性
CONTAINS SQL	表示子程序包含 SQL 语句，但不包含读写数据的语句
NO SQL	表示子程序不包含 SQL 语句
READS SQL DATA	表示子程序中包含读数据的语句
MODIFIES SQL DATA	表示子程序中包含写数据的语句
SQL SECURITY{DEFINER|INVOKER}	指明权限执行。DEFINER 表示只有定义者自己才能够执行；INVOKER 表示调用者可以执行
COMMENT'string'	是注释信息

例 15.11　下面应用此语句修改存储过程 count_of_student。（**实例位置：光盘\TM\Instance\15\15.11**）

其代码如下：

```
alter procedure count_of_student
modifies sql data
sql security invoker;
```

其运行结果如图 15.18 所示。

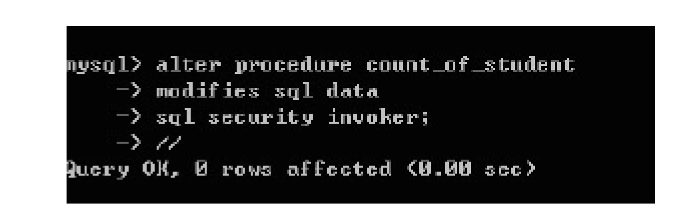


图 15.18　修改存储过程 count_of_student 的定义

技巧 如果读者希望查看修改后的结果，可以应用 SELECT FROM studentinfo.Ruotines WHERE ROUTINE_NAME='sp_name'来查看表的信息。由于篇幅限制，这里不能详细地讲解。

15.6　删除存储过程和函数

视频讲解：光盘\TM\Video\第 15 章\删除存储过程和函数.exe

删除存储过程和函数指删除数据库中已经存在的存储过程和函数。MySQL 中使用 DROP PROCEDURE 语句来删除存储过程，通过 DROP FUNCTION 语句来删除存储函数。在删除之前，必须确认该存储过程或函数没有任何依赖关系，否则可能会导致其他与其关联的存储过程无法运行。

删除存储过程和函数的语法如下：

```
DROP {PROCEDURE | FUNCTION} [IF EXISTS] sp_name
```

其中，sp_name 参数表示存储过程或函数的名称；IF EXISTS 是 MySQL 的扩展，判断存储过程或函数是否存在，以免发生错误。

例 15.12　下面删除名称为 count_of_student 的存储过程。（**实例位置：光盘\TM\Instance\15\15.12**）

其关键代码如下：

```
drop procedure count_of_student;
```

删除存储过程 count_of_student 的运行结果如图 15.19 所示。

例 15.13　下面删除名称为 name_of_student 的存储函数。（**实例位置：光盘\TM\Instance\15\15.13**）

其关键代码如下：

```
drop function name_of_student;
```

删除存储函数 name_of_student 的运行结果如图 15.20 所示。

```
mysql> drop procedure count_of_student//
Query OK, 0 rows affected (0.02 sec)

mysql>
```

图 15.19　删除 count_of_student 存储过程

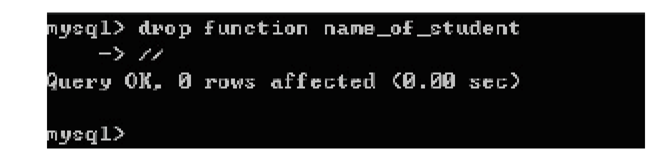


图 15.20　删除 name_of_student 存储函数

当返回结果没有提示警告或报错时，则说明存储过程或存储函数已经被顺利删除。用户可以通过查询 students 数据库下的 Routines 表来确认上面的删除是否成功。

15.7　捕获存储过程中的错误

视频讲解：光盘\TM\Video\第 15 章\捕获存储过程中的错误.exe

在用户执行某些程序时，可能会产生一些问题，为了增强程序本身处理问题的能力，避免程序因异常而终止运行。我们往往在处理程序执行前，预测程序在执行过程中可能出现或遇到的问题。定义条件和处理程序来提示用户的同时，也为用户提出解决办法。MySQL 通过 DECLARE 关键字来定义条件和处理程序。

15.7.1　定义条件

在 MySQL 中，应用 DECLARE 语句定义条件。其语法格式如下：

```
DECLARE condition_name CONDITION FOR condition_value
condition_value:
  SQLSTATE [VALUE] sqlstate_value | mysql_error_code
```

其中，condition_name 参数表示条件名称；condition_value 参数表示类型：sqlstate_value 参数可以表示 MySQL 的错误。另外也可以应用 mysql_error_code 来表示错误代码。其中第一种定义方法的格式如下：

```
DECLARE can_not_find CONDITION FOR SQLSTATE '42S02';
```

第二种定义格式如下：

```
DECLARE can_not_find CONDITION FOR 1146
```

说明 两种表示 MySQL 错误的方法名称都为 can_not_find，不同点只是获取的错误提示码不同。第一种方法设置 sqlstate_value 值为 42S02，第二种方法设置 mysql_error_code 值为 1146。

15.7.2　定义处理程序

另外，MySQL 中也可以使用 DECLARE 关键字来定义处理程序。其语法如下：

```
DECLARE handler_type HANDLER FOR condition_value[,...] sp_statement
handler_type:
 CONTINUE
 | EXIT
 | UNDO
condition_value:
  SQLSTATE [VALUE] sqlstate_value| condition_name   | SQLWARNING | NOT FOUND
 | SQLEXCEPTION| mysql_error_code
```

这个语句指定每个可以处理一个或多个条件的处理程序。如果产生一个或多个条件，指定的语句被执行。

☑ handler_type：指明错误的处理方式，该参数有 3 个取值。它们分别是：
  ➢ 对于一个 CONTINUE 处理程序，当前子程序的执行在执行处理程序语句之后继续。
  ➢ 对于 EXIT 处理程序，当前 BEGIN…END 复合语句的执行被终止。
  ➢ UNDO 表示遇到错误后撤回之前的操作，MySQL 不支持该处理方式。
☑ condition_value：指定错误类型，该参数有 6 个取值。
☑ sqlstate_value：表示 MySQL 的错误。
☑ mysql_error_code：表示错误代码
☑ condition_name： DECLARE 定义的条件名称。
☑ SQL WARNING：表示所有以 01 开头的 sqlstate_value 值。
☑ NOT FOUND：表示所有以 02 开头的 sqlstate_value 值。
☑ SQLEXCEPITION：表示所有没有被 SQL WARNING 或 NOT FOUND 捕获的 sqlstate_value 值。

15.8 实　　战

15.8.1 使用存储过程实现用户注册（PHP）

例 15.14 将向读者介绍 MySQL 5.5 版本中存储过程的创建以及 PHP 调用 MySQL 存储过程的方式。在图中的文本框中输入注册信息后，单击“注册”按钮即可将用户填写的注册信息保存到数据库中。（**实例位置：光盘\TM\Instance\15\15.14**）

实现过程如下所示。

（1）创建 pro_reg 存储过程，其代码如下：

```
delimiter //
create procedure pro_reg(in nc varchar(50),in pwd varchar(50),in email varchar(50),in address varchar(50))
begin
insert into tb_reg(name,pwd,email,address) values (nc,pwd,email,address);
end;
//
```

（2）通过 PHP 预定义类 mysqli 实现与 MySQL 数据库的连接。代码如下：

```
<?php
 if($_POST['submit']!=""){
    $conn=new mysqli("localhost","root","111","db_database15");                    //连接数据库
    $conn->query("set names utf8");                                                 //设置编码格式
    $name=$_POST['name'];
    $pwd=md5($_POST['pwd']);
    $email=$_POST['email'];
    $address=$_POST['address'];
```

（3）调用存储过程 pro_reg 实现将用户录入的注册信息保存到数据库中。代码如下：

```
  if($sql=$conn->query("call pro_reg('".$name."','".$pwd."','".$email."','".$address."')")){    //调用存储过程
          echo "<script>alert('用户注册成功!');</script>";
     }else{
           echo "<script>alert('用户注册失败!');</script>";
     }
 }
?>
```

运行结果如图 15.21～图 15.23 所示。

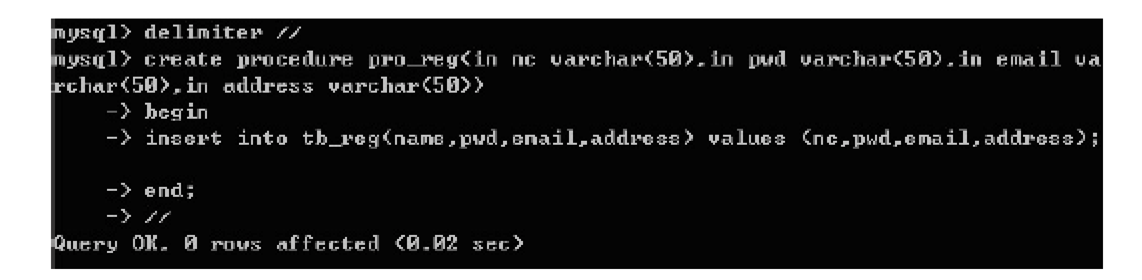


图 15.21　创建存储过程



图 15.22　录入注册信息



图 15.23　注册信息被存储到 MySQL 数据库

15.8.2　修改存储函数

例 15.15　将 name_of_student 的存储函数的读写权限修改为 READS SQL DATA，并加上注释信息 ‘FIND NAME’。修改存储过程和存储函数是指修改已经定义好的存储过程和函数。MySQL 中通过 ALTER PROCEDURE 语句来修改存储过程。（**实例位置：光盘\TM\Instance\15\15.15**）

代码如下：

```
ALTER FUNCTION name_of_student READS SQL DATA COMMENT ‘FIND NAME’;
SELECT SPCIFIC_NAME,SQL_DATA_ACCESS,ROUTINE_COMMENT FROM information_schema.Routines
WHERE ROUTINE_NAME='name_of_student';
```

运行结果如图 15.24 所示。

15.8.3　从 information_schema.Routines 表中查看存储过程

例 15.16　从 Routines 表中查询名为 count_of_student 的存储过程的信息。存储过程和函数的信息存储在 information_schema 数据库下的 Routines 表中，可以通过查询该表的记录来查询存储过程和函数的信息。（**实例位置：光盘\TM\Instance\15\15.16**）

关键代码如下：

```
SELECT * FROM information_schema.Routines WHERE ROUTINE_NAME='count_of_student' \G;
```

运行结果如图 15.25 所示。

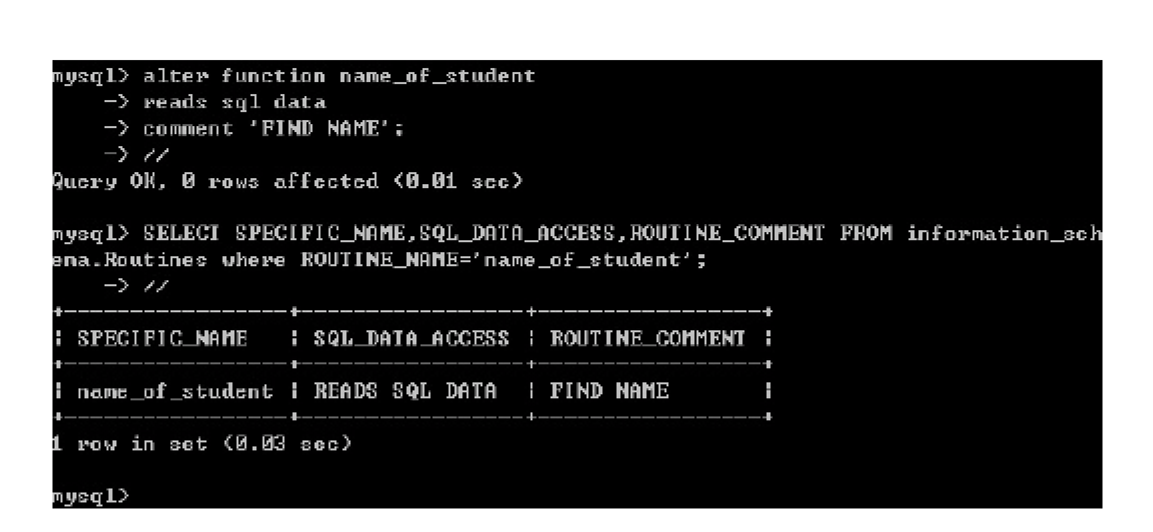


图 15.24　修改存储函数

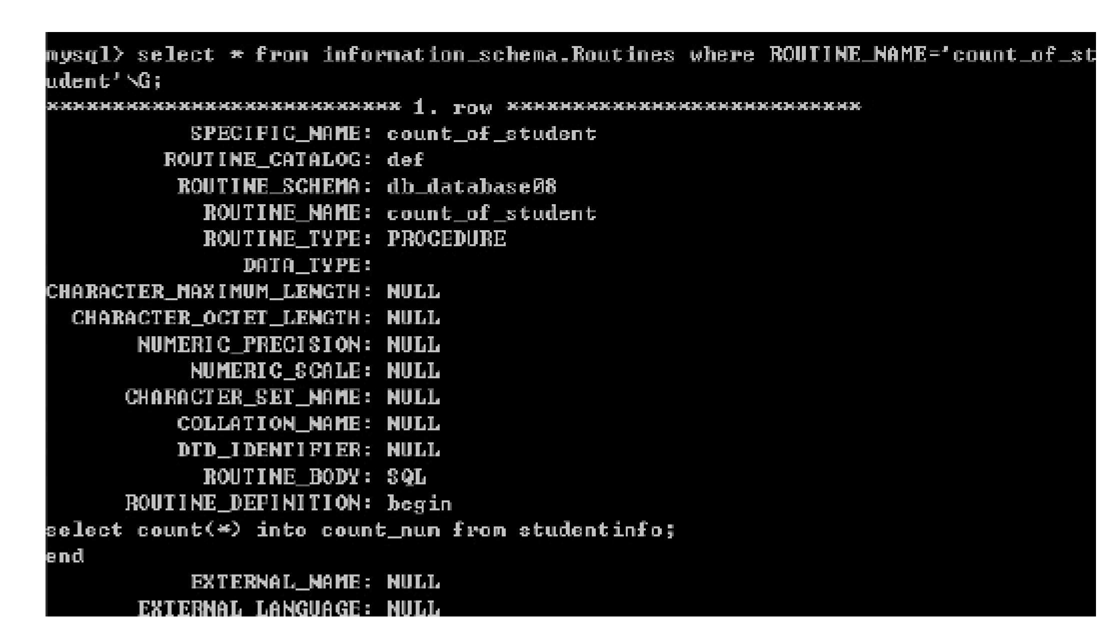
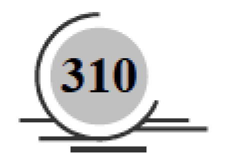


图 15.25　从 information_schema.Routines 表中查看存储过程



15.9　小　　结

本章对 MySQL 数据库的存储过程和存储函数进行了详细讲解，存储过程和存储函数都是用户自己定义的 SQL 语句的集合。它们都存储在服务器端，只要调用就可以在服务器端执行。本章重点讲解了创建存储过程和存储函数的方法，通过 CREATE PROCEDURE 语句来创建存储过程，通过 CREATE FUNCTION 语句来创建存储函数。需要读者将书中的知识点结合实际操作进行练习。

15.10　学习成果检验

1．存储过程和存储函数的区别是什么？（**实例位置：光盘\TM\Instance\15\15.17**）
2．一个存储过程中可以调用其他的存储过程吗？（**实例位置：光盘\TM\Instance\15\15.18**）
3．存储函数和 MySQL 内部函数有什么区别？（**实例位置：光盘\TM\Instance\15\15.19**）

第16章

MySQL 事务

（视频讲解：14 分钟）

在操作 MySQL 的过程中，对于一般的简单业务逻辑或中小型程序而言，无须考虑应用 MySQL 事务。但在比较复杂的情况下，在执行某些数据操作过程中，往往需要通过一组 SQL 语句执行多项并行业务逻辑或程序，这样，就必须保证所用命令执行的同步性，使执行序列中产生依靠关系的动作能够同时操作成功或同时返回初始状态。在此情况下，就需要用户优先考虑使用 MySQL 事务处理。

通过阅读本章内容，你可以：

- 了解 MySQL 事务
- 了解 MySQL 事务的存在周期
- 掌握如何创建事务
- 掌握 MySQL 事务的查询、提交和回滚
- 掌握 MySQL 事务行为
- 掌握 MySQL 事务的性能
- 掌握如何使用 MySQL 的伪事务

16.1　MySQL 事务概述

视频讲解：光盘\TM\Video\第 16 章\MySQL 事务概述.exe

在 MySQL 中，事务由单独单元的一个或多个 SQL 语句组成。在这个单元中，每个 MySQL 语句是相互依赖的。而整个单独单元作为一个不可分隔的整体，一旦单元中某条 SQL 语句执行失败或产生错误，整个单元将会回滚，所有受到影响的数据将返回到事务开始以前的状态；如果单元中的所有 SQL 语句均执行成功，则事务被顺利执行。

在现实生活中，事务处理数据的应用非常广泛，如网上交易、银行事务等。下面通过网上交易流程向读者展示事务的概念。

相信大多数用户都有过网上购物的体验，即用户登录某个大型购物网站，浏览该网站中所陈列的商品信息，将自己喜欢的商品放入购物车中，选购完毕后，需要对选购的商品进行在线支付，当用户对所选商品付款完毕后，通知商家发货。在此过程中，用户所付货款并未提交到商户手中，当用户收到货物之后，确认收货，商家才收到商品货款，整个交易过程才算完成。如果任何一步操作失败，则都会导致双方陷入尴尬的境界，试想当用户选购完商品并付款后，在发货过程中取消订单。这时商家并没有得到取消操作提醒，如果不应用事务处理，则在用户取消订单操作过程后，商家仍然继续将用户所订购的商品发送给用户，这会导致一些不愉快的争端。故在整个交易过程中，必须采用事务来对网上交易进行回滚操作。其流程如图 16.1 所示。

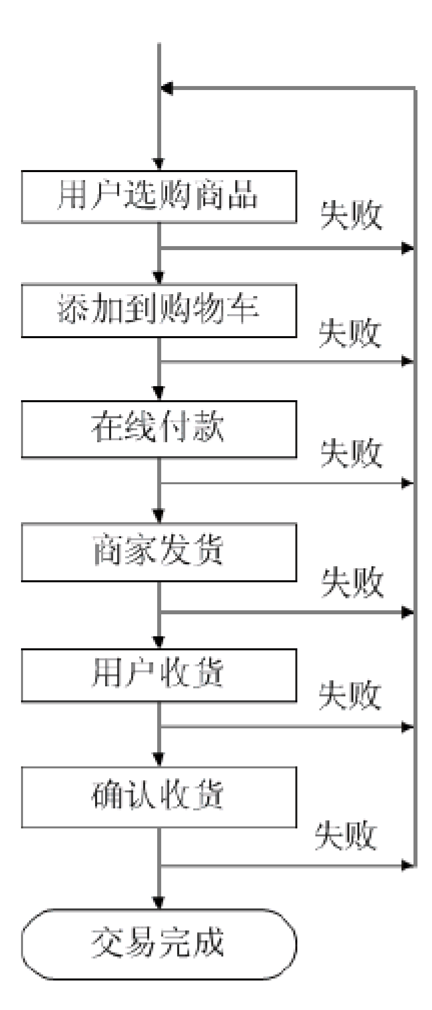
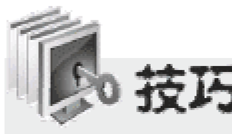


图 16.1　应用事务处理网上交易流程

在网上交易流程过程中，商家与用户的交易可以被认为是一个事务处理过程。其中整个事务过程中，如果存在一个环节失败，都可能导致双方交易的失败，如前面事务定义所说，所有这些流程都应该被成功执行。在 MySQL 中，如果其中有任何命令失败，都会导致所有操作命令被撤销，系统返回未操作前的状态，即回滚到初始状态。添加到购物车、在线付款、商家发货等构成了一个基本的事务。整个交易流程可以被看做一个完整的单元，用于实现整体事务。

通过 InnoDB 和 BDB 类型表，MySQL 事务能够完全满足事务安全的 ACID 测试。但是并不是所有表类型都支持事务，如 MyISAM 类型表就不能支持事务。只能通过伪事务对表实现事务处理。



ACID 指出每个事务型 RDBMS 必须遵守的 4 个属性，即原子性、一致性、孤立性和持久性。

16.1.1　原子性

原子性意味着事物的整体性和不可分割性，类似于化学中的原子，事务就具备这样的属性，其被认为是一个不可分割的单元。假设一个事务由多种任务组成，其中的语句必须同时操作成功，才可以认为事务是成功的，否则将回滚到初始状态。从网上交易的例子可以看出，所有操作的成功是保证交易完成的前提条件，当任何一个环节出现问题，即导致事务回滚，不能正常完成交易过程。即所有事务共同进退，保证了事务的整体性，这就是事务的原子性。

原子的执行是一个全部发生或全部失败的整体过程。在一个原子操作中，如果事务中的任何一个语句失败，前面执行的语句都将被返回，以保证数据的整体性不被破坏。这在常用的系统应用中，为保证数据的安全性起到一定作用。

16.1.2　一致性

在 MySQL 事务处理过程中，无论事务是完全成功还是在中途因某些环节失败而导致失败，事务使系统处于一致的状态时，其必须保证一致性。如在网上进行转账操作的过程中，用户 A 向用户 B 的账户中转入 5000 元，但用户 B 在查询转账信息时，发现自己的账户中只增加了 3000 元，这样不能使整个事务达到一致性。例如，从上述网上交易的例子考虑，当交易完成后，商家的货物会减少，不可能出现当商家给用户发货后货物数量不变，而用户收到货物后不增加这种情况。

在 MySQL 中，一致性主要由 MySQL 的日志机制处理，它记录数据库的所有变化，为事务回复提供跟踪记录。如果系统在事务处理中间发生错误，MySQL 恢复过程将使用这些日志发现事务是否已经完全成功执行或需要返回。一致性属性保证数据库从不返回一个未处理的事务。

16.1.3　孤立性

孤立性是指每个事务在自己的空间发生，和其他发生在系统中的事务隔离，而且事务的结果只在它完全被执行时才能看到。即使系统中同时发生多个事务，孤立性也可以保证特定的事务在完成之前，其结果是不被公布的。

当系统支持多个同时存在的用户和连接时，系统必须遵守孤立性原则，否则在执行过程中可能导致大量数据被破坏。孤立性保证每个事务完整的、在其各自的空间内被顺利执行，保证事务与事务之间不会相互冲突。

16.1.4　持久性

在 MySQL 中，即便是数据库系统崩溃，一个提交的事务仍然在坚持。当一个事务完成，数据库的日志已经被更新时，持久性即可发挥其特有功效。在 MySQL 中，如果系统崩溃或者数据存储介质被破坏，通过使用日志，系统能够恢复在重启前进行的最后一次成功更新，可以反映系统崩溃时处于执行过程的事务的变化。

MySQL 的持久性是通过一条记录事务过程中系统变化的二进制事务日志文件来实现的。如果遇到硬件损坏或者系统异常关机，系统在下一次启动时，通过使用最后的备份和日志就可以恢复丢失的数据。



默认情况下，InnoDB 表持久性最久，MyISAM 表具有部分持久性。

16.2　MySQL 事务的创建与存在周期

视频讲解：光盘\TM\Video\第 16 章\MySQL 事务的创建与存在周期.exe

通过上述对事务定义的叙述和事务特性的讲解，相信读者已经对事务有一个初步的认识。下面将对事



务的存在周期做详细讲解。首先，向读者介绍如何在 MySQL 中创建事务。

创建事务的一般过程是初始化事务、创建事务、应用 SELECT 语句查询数据是否被录入和提交事务。如果用户不在操作完数据库后执行事务提交，则系统会默认执行回滚操作。如果用户在提交事务前选择撤销事务，则用户在撤销前的所有事务将被取消。数据库系统会回到初始状态。

注意　默认情况下，在 MySQL 中创建的数据表类型都是 MyISAM，但是该类型的数据表并不能支持事务。所以，如果想让数据表支持事务处理能力，必须将当前操作数据表的类型设置为 InnoDB 或 BDB。

在创建事务的过程中，需要创建一个 InnoDB 或 BDB 类型的数据表。其基本命令结构如下：

```
CREATE TABLE    table_name(field_defintions) TYPE = INNODB/BDB;
```

其中，table_name 为表名；field_defintions 为表内定义的字段等属性；TYPE 可指定数据表的类型，既可以是 InnoDB 类型，可以是 BDB 类型。

如果希望让已经存在的表支持事务处理，则可以应用 ALTER TABLE 命令指定数据表的类型，即可实现对表类型的更改操作，使原本不支持事务的数据表更改为支持事务处理的类型。其命令如下：

```
ALTER TABLE    table_name TYPE= INNODB/BDB;
```

更改完表的类型后，即可使数据表支持事务处理。

技巧　应用 ALTER TABLE 操作可能会导致数据库中的数据丢失，因此为了避免非预期结果出现，在使用 ALTER TABLE 命令前，需要创建一个表备份。

16.2.1　初始化事务

初始化 MySQL 事务，首先声明初始化 MySQL 事务后所有的 SQL 语句为一个单元。在 MySQL 中，应用 START TRANSACTION 命令来标记一个事务的开始。初始化事务的结构如下：

```
START TRANSACTION;
```

另外，也可以使用 BEGIN 或者 BEGIN WORK 命令初始化事务，通常 START TRANSACTION 命令后面跟随的是组成事务的 SQL 语句。

在命令提示符中输入如下命令：

```
start transaction;
```

如果输入以上代码后，MySQL 数据库没有给出警告提示或返回错误信息，则说明事务初始化成功，可以继续执行下一步操作。

16.2.2　创建事务

例 16.1　初始化事务成功后，可以创建事务。这里以向名称为 connection 的数据表中插入一条记录为例，讲解事务的创建。首先打开数据库，选定某个数据库，然后初始化事务，最后创建事务，向指定的数据表中添加记录。**（实例位置：光盘\TM\Instance\16\16.1）**

其代码如下：

```
start transaction;
insert into connection(email,cellphone,QQ,sid)
values('mrsoft@mrsoft.com',7856456789,92343432,3);
```

其运行结果如图 16.2 所示。

16.2.3　应用 SELECT 语句查看数据是否被正确输入

事务创建成功后，建议读者通过 SELECT 语句查看数据是否被正确输入。

例 16.2　在事务初始化成功后创建事务。**（实例位置：光盘\TM\Instance\16\16.2）**

在命令提示符中输入如下指令：

```
SELECT * FROM connection WHERE   sid=3;
```

其运行结果如图 16.3 所示。

```
mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into connection(email,cellphone,QQ,sid) values('mrsoft@mrsoft.com'
,7856456789,92343432,3);
Query OK, 1 row affected, 1 warning (0.00 sec)
```

图 16.2　创建事务

```
mysql> select * from connection where sid=3;
+------+-------------------+------------+----------+------+
| id   | email             | cellphone  | QQ       | sid  |
+------+-------------------+------------+----------+------+
| NULL | mrsoft@mrsoft.com | 2147483647 | 92343432 |    3 |
+------+-------------------+------------+----------+------+
1 row in set (0.00 sec)
```

图 16.3　查看数据是否被正确输入

技巧 在插入新表为 InnoDB 类型或更改原来表类型为 InnoDB 时，如果在输入命令提示后，MySQL 提示“The 'InnoDB' feature is disabled; you need InnoDB ' to have it working”警告，则说明 InnoDB 表类型并没有被开启，用户需要找到 MySQL 文件目录下的 my.ini 文件，定位 skip_innodb 选项位置，将原来的 skip_ innodb 改为#skip_innodb 后保存该文件，重新启动 MySQL 服务器，即可令数据库支持 InnoDB 类

16.2.4　提交事务

在用户没有提交事务之前，当其他用户连接 MySQL 服务器时，应用 SELECT 语句查询结果，则不会显示没有提交的事务。当且仅当用户成功提交事务后，其他用户才可以通过 SELECT 语句查询事务结果。由事务的特性可知，事务具有孤立性，当事务处在处理过程中时，其实 MySQL 并未将结果写入磁盘中，这样一来，这些正在处理的事务相对其他用户是不可见的。一旦数据被正确插入，用户可以使用 COMMIT 命令提交事务。提交事务的命令结构如下：

```
COMMIT
```

一旦当前执行事务的用户提交当前事务，则其他用户就可以通过会话查询结果。

16.2.5　撤销事务（事务回滚）

撤销事务，又被称做事务回滚，即事务被开启、用户输入的 SQL 语句被执行后，如果用户想要撤销刚才的数据库操作，可使用 ROLLBACK 命令撤销数据库中的所有变化。ROLLBACK 命令的结构如下：

```
ROLLBACK
```

输入回滚操作后，可以通过 SELECT 语句查看是否执行了回滚操作。例如，可以通过 SELECT 语句查看 16.2.2 节中插入的数据是否存在，其运行结果如图 16.4 所示。

注意 如果执行一个回滚操作，则在 START TRANSACTION 命令后的所有 SQL 语句都将执行回滚操作。故在执行事务回滚前，用户需要慎重选择是否执行回滚操作。如果开启事务后，没有提交事务，则事务默认为自动回滚状态，即不保存用户之前的任何操作。

技巧 在现实应用中，事务撤销即事务回滚有着很重要的意义。例如，用户 A 和用户 B 采用银行转账方式交易，用户 A 将个人账户的部分存款转移到用户 B 的个人账户过程中，若银行的数据库系统突然发生错误或异常，则交易事务提交失败，系统执行回滚操作，恢复到交易的初始状态。这样可以避免因特殊情况而导致事务提交失败，从而减少不必要的损失。

16.2.6　事务的存在周期

事务的周期由用户在命令提示符中输入 START TRANSACTION 指令开始，直至用户输入 COMMIT 结束，如图 16.5 所示为一个简单事务存在周期流程图。

```
mysql> select * from connection where sid=3;
Empty set (0.00 sec)

mysql>
```

图 16.4　执行回滚操作后应用 SELECT 查询

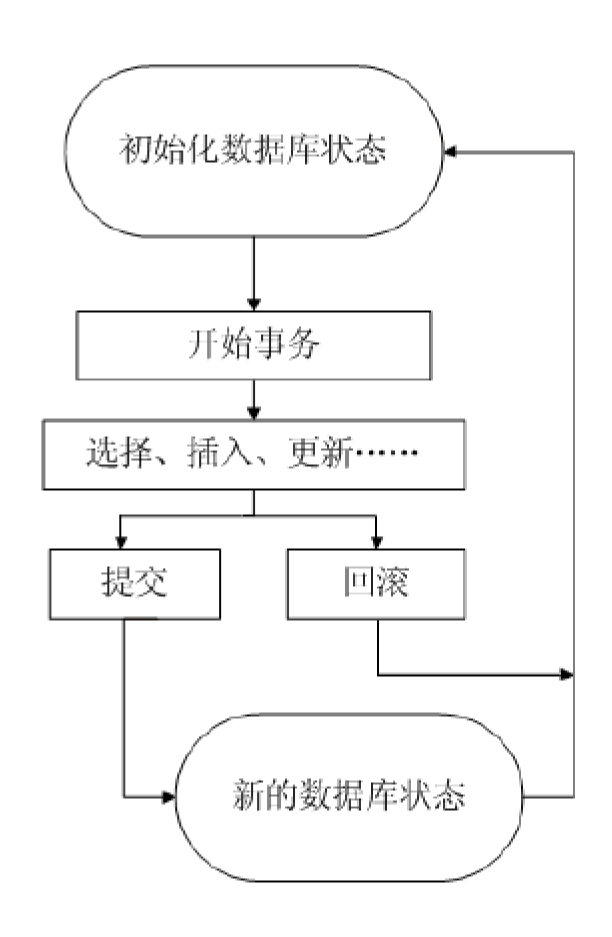


图 16.5　事务的存在周期

技巧 事务不支持嵌套功能，当第一个事务未结束又重新打开一个事务时，则前一个事务会自动提交。同样，MySQL 命令中很多命令都会隐藏执行 COMMIT 命令。

16.3　MySQL 行为

在 MySQL 中，存在两个可以控制行为的变量，分别是 AUTOCOMMIT 变量和 TRANSACTION ISOLACTION LEVEL 变量。

16.3.1　自动提交

在 MySQL 中，如果不更改其自动提交变量，系统会自动向数据库提交结果。用户在执行数据库操作过程中，不需要使用 START TRANSACTION 语句开始事务，应用 COMMIT 或者 ROLLBACK 提交事务或执行回滚操作。如果用户希望通过控制 MySQL 自动提交参数，可以更改提交模式，这一更改过程是通过设置 AUTOCOMMIT 变量来实现的。

下面通过一个实例向读者展示如何关闭自动提交功能。在命令提示符中输入以下命令：

```
SET AUTOCOMMIT=0 ;
```

此时即可关闭自动提交功能。只有当用户输入 COMMIT 命令后，MySQL 才将数据表中的资料提交到数据库中，如果不提交事务，而终止 MySQL 会话，数据库将会自动执行回滚操作。

例 16.3 在关闭自动提交功能后，查询数据表中数据，并且向数据表中添加一条记录。**（实例位置：光盘\TM\Instance\16\16.3）**

其命令如下：

```
set autocommit=0;
select * from timeinfo;
insert into timeinfo(info) values('test autocommit');
```

以上代码的运行结果如图 16.6 所示。

由于用户已经关闭自动提交功能，所以图 16.6 中所示的添加操作因没有执行事务的提交操作，而导致数据没有成功添加。再次查询数据表中的数据，运行结果如图 16.7 所示。

```
mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from timeinfo;
Empty set (0.00 sec)

mysql> insert into timeinfo(info) values('test autocommit');
Query OK, 1 row affected (0.00 sec)

mysql> exit
Bye
```

图 16.6 取消自动提交操作

```
mysql> select * from timeinfo;
Empty set (0.00 sec)
```

图 16.7 应用 SELECT 查询自动提交关闭后的数据

结果表明，之前插入的数据并未插入到数据库中。另外，我们可以通过查看“@@AUTOCOMMIT”变量来查看当前自动提交状态，查看此变量同样应用 SELECT 语句，其运行结果如图 16.8 所示。



图 16.8 查看自动提交变量

16.3.2 事务的孤立级

事务具有独立的空间，在 MySQL 服务器中，用户通过不同的会话执行不同的事务，在多用户环境中，许多 RDBMS 会话在任意指定时刻都是活动的。为了使这些事务互不影响，保证数据库性能不受到影响，采用事务的孤立级是十分有必要的。

孤立级在整个事务中起到了很重要的作用，如果没有事务的孤立性，不同的 SELECT 语句将会在同一事务的环境中检索到不同的结果，这将导致数据的不一致性，给不同的用户造成困扰，这样一来，用户就不能以查询的结果集作为计算基础。所以孤立性强制保持每个事务的独立性，以此来保证事务中看到一致的数据。

基于 ANSI/ISO SQL 规范，MySQL 提供了以下 4 种孤立级。

☑ SERIALIZABLE（序列化）：顾名思义，以序列的形式对事务进行处理，该孤立级的特点是只有当事务提交后，用户才能从数据库中查看数据的变化。该孤立级运行会影响 MySQL 的性能，因为需要占用大量资源，以保证使大量事务在任意时间不被用户看到。

☑ REPEATABLE READ（可重读）：对于应用程序的安全性做出部分妥协，以提高其性能。事务在该孤立级上不会被看成一个序列，不过当前在执行事务的过程中，用户仍然看不到事务的过程。直到事务提交为止，用户才能够看到事务的变化结果。

☑ READ COMMITTED（提交后读）：提交后读孤立级的安全性比重复读安全性要低。在这一级的事务，用户可以看到其他事务添加的新记录。在事务处理时，如果存在其他用户同时对事务的相应表进行修改，那么在同一事务中不同时间内，应用 SELECT 语句可能返回不同的结果集。

☑ READ UNCOMMITTED（未提交读）：该孤立级提供事务之间的最小程度间隔，容易产生虚幻读

操作。其他用户可以在该孤立级上看到未提交的事务。

16.3.3　修改事务的孤立级

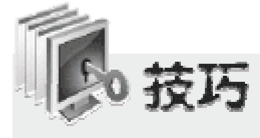
在 MySQL 中，可以使用 TRANSACTION ISOLATION LEVEL 变量修改事务孤立级，其中，MySQL 的默认孤立级为 REPEATABLE READ（可重读），用户可以使用 SELECT 命令获取当前事务孤立级变量的值，其命令如下：

```
SELECT @@tx_isolation;
```

例 16.4　如果希望修改事务的孤立级，可以通过 SET 命令设置其不同值，来修改事务的孤立级，操作命令如图 16.9 所示。（**实例位置：光盘\TM\Instance\16\16.4**）



图 16.9　设置事务孤立级



如果想修改事物的孤立级，必须首先获取 SUPER 优先权，以便可以顺利执行修改操作。

16.4　事务和性能

从 16.3 节中可以看出，应用不同孤立级的事务可能会对系统造成一系列影响，采用不同孤立级处理事务，可能会对系统稳定性和安全性等诸多因素造成影响。另外，有些数据库操作中，不需要应用事务处理，则用户需要选择合适的数据表类型。在选择表类型时，应该考虑数据表具有完善的功能，且高效执行时不会对系统增加额外的负担。

16.4.1　应用小事务

应用小事务的意义在于：保证每个事务不会在执行前等待很长时间，从而避免因为各个事务互相等待而导致系统的性能大幅度下降。用户在应用少数大事务时，可能无法看出因事务间互相等待而导致系统性能下降，但是当系统中存在处理量很大的数据库或多种复杂事务时，因长时间等待而导致系统性能下降就很明显。所以，应用小事务可以保证系统的性能，小事务可以快速变化或退出，这样，其他在队列中准备就绪的事务就不会受到明显影响。

16.4.2　选择合适的孤立级

事务的性能与其对服务器产生的负载成反比，即当事务孤立级越高，其性能越低，但是其安全性也越高。事务性能和孤立级关系如图 16.10 所示。所以只有选择适当的孤立级，才能有效地提高 MySQL 系统性能和应用性。

图 16.10　事务孤立级和性能的关系

从图 16.10 可以看出，虽然稳定性随着孤立级的增高而改变，但是并不代表孤立级的稳定性越高，其灵活性越高，故用户在选择孤立级时，需要根据实际情况选择适合应用的孤立级，切勿生搬硬套。

16.4.3　死锁的概念与避免

死锁，即当两个或者多个处于不同序列的用户打算同时更新某相同的数据库时，因互相等待对方释放权限而导致双方一直处于等待状态。在实际应用中，两个不同序列的客户打算同时对数据执行操作，极有可能产生死锁。更具体地讲，当两个事务相互等待操作对方释放所持有的资源，而导致两个事务都无法操作对方持有的资源，这样无限期的等待被称做死锁。

不过，MySQL 的 InnoDB 表处理程序具有检查死锁功能，如果该处理程序发现用户在操作过程中产生死锁，将立刻通过撤销操作来撤销其中一个事务，以便使死锁消失。这样就可以使另一个事务获取对方所占有的资源而执行逻辑操作。

16.5　MySQL 伪事务

在 MySQL 中，InnoDB 和 BDB 类型表可以支持事务处理，但是 MyISAM 类型表并不能支持事务处理，对于该类型表，用户可以选择应用表锁定来替代事务。这种引用表锁定来替代事务的事件被称做伪事务。使用表锁来锁定表的操作，可以加强非事务表在执行过程的安全性和稳定性。

16.5.1　用表锁定代替事务

在 MySQL 的 MyISAM 类型数据表中，并不支持 COMMIT（提交）和 ROLLBACK（回滚）命令。当用户对数据库执行插入、删除、更新等操作时，这些变化的数据都被立刻保存在磁盘中。这样，在多用户环境中会导致诸多问题，为了避免同一时间有多个用户对数据库中指定表进行操作，可以应用表锁定来避免在用户操作数据表过程中受到干扰。当且仅当该用户释放表的操作锁定后，其他用户才可以访问这些修改后的数据表。

设置表锁定代替事务的基本步骤如下。

（1）为指定数据表添加锁定。其语法如下：

```
LOCK TABLES table_name lock_type,…
```

其中，table_name 为被锁定的表名；lock_type 为锁定类型，该类型包括以读方式（READ）锁定表、以写方式（WRITE）锁定表。

（2）执行数据表的操作，可以添加、删除或者更改部分数据。

（3）完成对锁定数据表的操作后，需要对该表进行解锁操作，释放该表的锁定状态。其语法如下：

```
UNLOCK TABLES
```

下面通过实例向读者展示如何以读方式和以写方式锁定数据表。

1. 以读方式锁定数据表

以读方式锁定数据表是设置锁定用户的其他方式操作，如删除、插入、更新都不被允许，直至用户进行解锁操作。

例 16.5　锁定 studentinfo 数据表。

首先，在命令提示符下输入如下代码：**（实例位置：光盘\TM\Instance\16\16.5）**

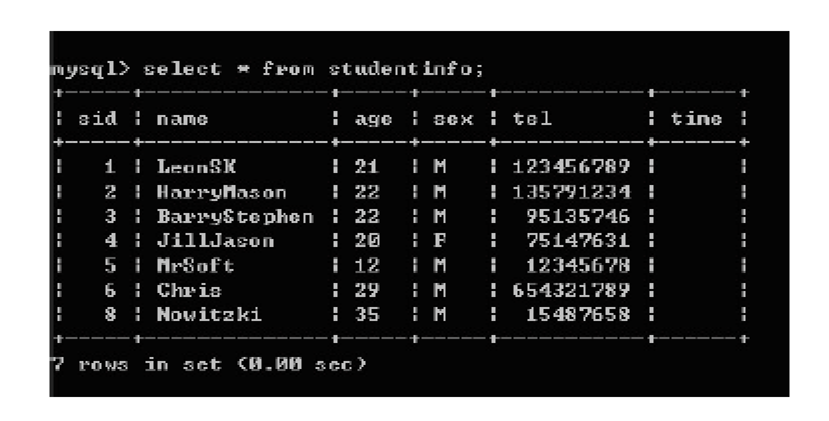
```
lock table studentinfo read;
```



然后，应用 SELECT 语句查看表中的信息，其运行结果如图 16.11 所示。

下面，尝试向该数据表中插入一条数据，其运行结果如图 16.12 所示。

从上述结果可以看出，当用户试图向数据库插入数据时，将会返回失败信息。当将锁定的表解锁后，再次执行插入操作，其运行结果如图 16.13 所示。



```
mysql> select * from studentinfo;
+-----+---------------+-----+-----+-----------+------+
| sid | name          | age | sex | tel       | time |
+-----+---------------+-----+-----+-----------+------+
|   1 | LeonSK        | 21  | M   | 123456789 |      |
|   2 | HarryMason    | 22  | M   | 135791234 |      |
|   3 | BarryStephen  | 22  | M   |  95135746 |      |
|   4 | JillJason     | 20  | F   |  75147631 |      |
|   5 | MrSoft        | 12  | M   |  12345678 |      |
|   6 | Chris         | 29  | M   | 654321789 |      |
|   8 | Nowitzki      | 35  | M   |  15487658 |      |
+-----+---------------+-----+-----+-----------+------+
7 rows in set (0.00 sec)
```

图 16.11　查看以读方式锁定的 studentinfo 表

```
mysql> insert into studentinfo (name,age,sex,tel)values
    -> ('LarryBird','29','M',1874930203);
ERROR 1100 (HY000): Table 'timeinfo' was not locked with LOCK TABLES
mysql>
```

图 16.12　向以读方式锁定的表中插入数据

```
mysql> unlock tables;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into studentinfo (name,age,sex,tel)values
    -> ('LarryBird','29','M',1874930203);
Query OK, 1 row affected, 1 warning (0.08 sec)
```

图 16.13　向解锁后的数据表中添加数据

锁定被释放后，用户可以对数据库执行添加、删除、更新等操作。

技巧　其中，lock_type 参数中，用户指定数据表以读方式(READ)锁定数据表的变体为 READ LOCAL 锁定。其与 READ 锁定的不同点是，该参数所指定的用户会话可以执行 INSERT 操作。它是为了使用 MySQL dump 工具而创建的一种变体形式。

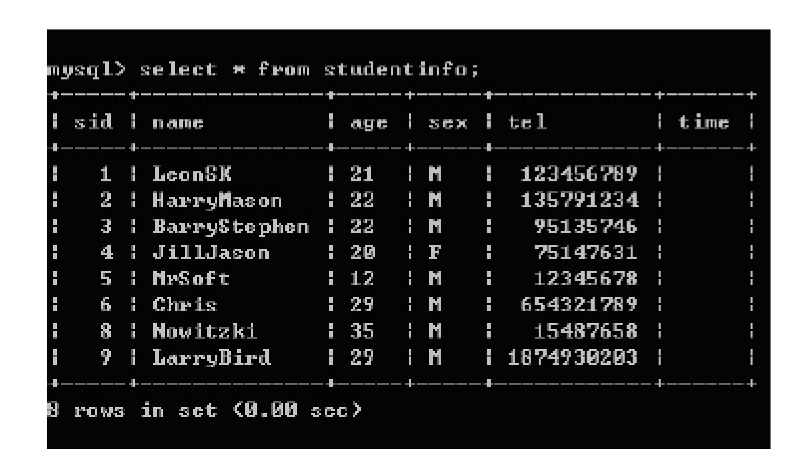
2. 以写方式锁定数据表

与读方式锁定表类似，表的写锁定是设置用户可以修改数据表中的数据，但是除自己以外，会话中的其他用户不能进行任何读操作。在命令提示符中输入如下命令：

```
lock table studentinfo write;
```

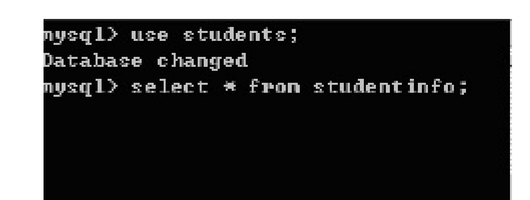
例 16.6　因为该表为写锁定，用户可以对数据库的数据执行修改、添加、删除等操作。在该命令提示符中应用 SELECT 语句查询该锁定表，其运行结果如图 16.14 所示。**（实例位置：光盘\TM\Instance\16\16.6）**

从图 16.14 中可以看到，当前用户应用 SELECT 语句仍然可以查询该表的数据，并没有限制用户对数据表的读操作。这是因为，以写方式锁定数据表并不能限制当前锁定用户的查询操作。下面打开一个新用户会话，即保持图 16.14 所示窗口不被关闭，重新打开一个新的 MySQL 连接，并执行上述过程。其运行结果如图 16.15 所示。



```
mysql> select * from studentinfo;
+-----+---------------+-----+-----+------------+------+
| sid | name          | age | sex | tel        | time |
+-----+---------------+-----+-----+------------+------+
|   1 | LeonSK        | 21  | M   |  123456789 |      |
|   2 | HarryMason    | 22  | M   |  135791234 |      |
|   3 | BarryStephen  | 22  | M   |   95135746 |      |
|   4 | JillJason     | 20  | F   |   75147631 |      |
|   5 | MrSoft        | 12  | M   |   12345678 |      |
|   6 | Chris         | 29  | M   |  654321789 |      |
|   8 | Nowitzki      | 35  | M   |   15487658 |      |
|   9 | LarryBird     | 29  | M   | 1874930203 |      |
+-----+---------------+-----+-----+------------+------+
8 rows in set (0.00 sec)
```

图 16.14　查询应用写操作锁定的 studentinfo 数据表



```
mysql> use students;
Database changed
mysql> select * from studentinfo;
```

图 16.15　打开新会话查询被锁定的数据表

在新打开的命令提示界面可以看到，应用 SELECT 语句执行查询操作，并没有结果显示，这是因为之前该表以写方式锁定。故当操作用户释放该数据表锁定后，其他用户才可以通过 SELECT 语句查看之前被锁定的数据表。在命令提示符中输入如下代码：

```
UNLOCK TABLES;
```

这时，在第二次打开的命令提示符中，即可出现如图 16.14 所示的结果。当数据表被释放锁定后，其他访问数据库的用户即可查看数据表的内容。


技巧 使用 UNLOCK TABLE 命令后，将会释放所有当前处于锁定状态的数据表。

16.5.2 应用表锁实现伪事务

相信读者通过上面的学习已经了解什么是表锁，下面通过使用表锁对 MyISAM 表进行锁定操作，以此过程来代替事务型表 InnoDB，即应用表锁来实现伪事务。

实现伪事务的一般步骤如下。

（1）对数据库中的数据表进行锁定操作，可以对多个表做不同的方式锁定。其代码格式如下：

```
LOCK TABLE table_name1 lock_type1, table_name2 lock_type2,...
```

（2）执行数据库操作，向锁定的数据表中执行添加、删除、修改操等操作。如前面提到的 INSERT、UPDATE、DELETE 等操作。用户可以对锁定的数据表执行上述操作，在执行过程中，该伪事务所产生的结果是不会被其他用户更改的。

（3）释放锁定的数据表，以便让正在队列中等待查看或操作的其他用户可以浏览数据表中的数据或对操作表执行各种数据的操作。

如果存在其他会话要求访问已锁定的多个表格，则该会话必须被迫等待当前锁定用户释放锁定表，才允许其他会话访问该数据表，表锁定使不同会话执行的数据库操作彼此独立。应用数据表锁定方式可以使不支持事务类型的表实现伪事务。

16.6 实　　战

16.6.1 使用事务处理技术实现银行的安全转账（PHP）

例 16.7 在实现银行转账过程中，发生意外是在所难免的，为了避免因意外而造成不必要的损失，在银行转账时经常使用事务处理方式。运行本实例，效果如图 16.16 所示，在文本框中输入要转给 B 账户的金额后，单击“转账”按钮即可实现转账。（**实例位置：光盘\TM\Instance\16\16.7**）



图 16.16　银行转账

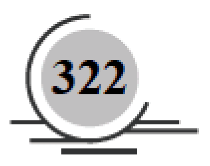
实现过程如下所示。

（1）利用 mysqli 扩展技术实现与 MySQL 数据库的连接，代码如下：

```
<?php
$conn=new mysqli("localhost","root","111","db_database16");          //连接数据库
$conn->query("set names utf8");                                      //设置编码格式
?>
```

（2）查询 A 账户与 B 账户的现有金额并显示在页面中。代码如下：

```
include_once("conn.php");                                            //包含数据库连接文件 conn.php
    $sql=$conn->query("select * from tb_zy where flag='mrsoft'");    //查询 mrsoft 用户的信息
    $info=$sql->fetch_array(MYSQLI_ASSOC);
    $sql1=$conn->query("select * from tb_zy where flag='mr'");       //查询 mr 用户的信息
    $info1=$sql1->fetch_array(MYSQLI_ASSOC);
```



（3）实现转账，其主要代码如下：

```
$tob=$_POST[tob];                                                    //接收提交的转账金额
include_once("conn.php");                                            //包含数据库连接文件 conn.php
$conn->autocommit(false);                                            //取消查询自动提交
if(!$conn->query("update tb_zy set money=money-'".$tob."' where flag='mrsoft'")){ //减少指定账户所要转账的金额
    $conn->rollback();                                               //如果失败则事务回滚
}
if(!$conn->query("update tb_zy set money=money+'".$tob."' where flag='mr'")){ //增加指定账户所要的转账金额
    $conn->rollback();                                               //如果失败则事务回滚
}
$conn->commit();                                                     //提交查询
$conn->autocommit(true);                                             //恢复查询自动提交
echo "<script>window.location.href='index.php';</script>";           //重定向到首页
```

16.6.2　批处理中使用事务（Java）

例 16.8　在企业级的应用程序中经常会遇到对多个数据表同时存取的情况，最明显的例子是银行的转账业务，从汇款账户中减去指定金额，并将该金额添加至收款账户中，但如果在转账的过程中发生程序错误或者系统断电等意外情况，就可能导致汇款账户的余额已经减少而收款账户的余额还没有增加，这就需要应用事务对该问题进行处理。本实例模拟银行转账系统，通过事务保证转账业务的顺利进行。**（实例位置：光盘\TM\Instance\16\16.8）**

实现过程如下所示。

（1）在项目中创建 BatchAffair 类，在该类中定义操作数据的各种方法，其中定义获取账户表 tb_transition 中所有账户方法 selectIds()，该方法以 List 集合对象作为返回值。具体代码如下：

```
public List selectIds() {
    conn = getConn();                                                //获取数据库连接
    Statement cs = null;                                             //定义 CallableStatement 对象
    String sql = "Select accoutNumber from tb_transition";           //定义查询视图的 SQL 语句
    List list = new ArrayList();                                     //定义保存查询结果的 List 集合
    try {
        cs = conn.createStatement();                                 //实例化 Statement 对象
        ResultSet rest = cs.executeQuery(sql);                       //执行 SQL 语句
        while (rest.next()) {                                        //循环遍历查询结果集
            String accoutNumber = rest.getString(1);
            list.add(accoutNumber);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return list;
}
```

（2）在 BatchAffair 类中定义转账方法 Batch()，该方法包含两个 String 类型的参数与一个 float 类型的参数，分别用于指定转账的账户、转入的账户和转账的金额。具体代码如下：

```
public void Batch(String incomeId, String goId, float money) throws SQLException {
    try {
        conn = getConn();                                            //获取数据库连接
        boolean autoCommit = conn.getAutoCommit();
        conn.setAutoCommit(false);

        Statement cs = null;                                         //定义 Statement 对象
```

```
            cs = conn.createStatement();                                    //实例化 Statement 对象
            cs.addBatch("update tb_transition set deposit = deposit-" + money
                        + " ,transition = transition-" + money
                        + " where accoutNumber = " + gold);                 //定义修改转账表中的数据方法
            cs.addBatch("update tb_transition set deposit = deposit+" + money
                        + " ,shift = shift+" + money + " where accoutNumber = "
                        + incomeId);
            cs.executeBatch();                                              //批量执行 SQL 语句
            cs.close();                                                     //将 Statement 对象关闭
            conn.commit();
            conn.setAutoCommit(autoCommit);
            conn.close();
        } catch (Exception e) {
            conn.rollback();
            e.printStackTrace();
        }
}
```

运行结果如图 16.17 所示。



图 16.17　银行转账业务

16.7　小　　结

本章对 MySQL 中事务的创建、提交、撤销，以及事务的存在周期进行了详细讲解，并通过举例说明，使读者更好地理解所学知识的用法。在阅读本章时，读者应该重点掌握如何自动提交事务和修改事务的孤立级。同时，应该了解 MySQL 事务和性能，以及 MySQL 伪事务。

16.8　学习成果检验

1．如何初始化事务？**（实例位置：光盘\TM\Instance\16\16.9）**
2．MySQL 中可以控制行为的变量为哪两个？**（实例位置：光盘\TM\Instance\16\16.10）**
3．如何设置表锁定代替事务？**（实例位置：光盘\TM\Instance\16\16.11）**

第17章

触发器

（视频讲解：21 分钟）

触发器是由事件来触发某个操作。这些事件包括 INSERT 语句、UPDATE 语句和 DELETE 语句。当数据库系统执行这些事件时，就会激活触发器执行相应的操作。本章将介绍触发器的含义和作用，可以了解创建触发器、查看触发器和删除触发器的方法。同时，还可以了解各种事件的触发器的执行情况。

通过阅读本章内容，你可以：

- 了解 MySQL 触发器的概念
- 了解在 MySQL 中创建单个执行语句的触发器
- 掌握 MySQL 中创建多个语句的触发器
- 掌握在 MySQL 数据库中查看触发器
- 掌握删除触发器的方法
- 掌握如何应用触发器

17.1 MySQL 触发器

视频讲解：光盘\TM\Video\第 17 章\MySQL 触发器.exe

触发器是由 MySQL 的基本命令事件来触发某种特定操作，这些基本的命令由 INSERT、UPDATE、DELETE 等事件来触发某些特定操作。满足触发器的触发条件时，数据库系统就会自动执行触发器中定义的程序语句。这样可以令某些操作之间的一致性得到协调。

17.1.1 创建 MySQL 触发器

在 MySQL 中，创建只有一个执行语句的触发器的基本形式如下：

```
CREATE  TRIGGER  触发器名  BEFORE | AFTER  触发事件
ON  表名  FOR EACH ROW  执行语句
```

具体的参数说明如下：

- ☑ 触发器名指定要创建的触发器名字。
- ☑ BEFORE 和 AFTER 指定触发器执行的时间。BEFORE 指在触发时间之前执行触发语句；AFTER 表示在触发时间之后执行触发语句。
- ☑ 触发事件指定数据库操作触发条件，其中包括 INSERT\UPDATE 和 DELETE。
- ☑ 表名指定触发时间操作表的名称。
- ☑ FOR EACH ROW 表示任何一条记录上的操作满足触发事件都会触发该触发器。
- ☑ 执行语句指定触发器被触发后执行的程序。

例 17.1 下面创建一个由插入命令“INSERT”触发的触发器 auto_save_time。**（实例位置：光盘\TM\Instance\17\17.1）**

具体步骤如下：

（1）创建一个名称为 timelog 的表格，该表的结构非常简单。相关代码如下：

```
create table timelog(
id int(11) primary key auto_increment not null,
savetime varchar(50) not null
);
```

（2）创建名称为 auto_save_time 的触发器。其代码如下：

```
delimiter //
create trigger auto_save_time before insert
on studentinfo for each row
insert into timelog(savetime) values(now());
//
```

以上代码的运行结果如图 17.1 所示。

auto_save_time 触发器创建成功，其具体的功能是当用户向 studentinfo 表中执行 INSERT 操作时，数据库系统会自动在插入语句执行之前向 timelog 表中插入当前时间。下面通过向 studentinfo 表中插入一条信息来查看触发器的作用。其代码如下：

```
insert into studentinfo(name) values ('Chris');
```

然后执行 SELECT 语句查看 timelog 表中是否执行 INSERT 操作，其结果如图 17.2 所示。

以上结果显示，在向 studentinfo 表中插入数据时，savetime 表中也会被插入一条当前系统时间的数据。

```
mysql> delimiter //
mysql> create trigger auto_save_time before insert
    -> on studentinfo for each row
    -> insert into timelog(savetime) values(now());
    -> //
Query OK, 0 rows affected (0.00 sec)
```

图 17.1　创建 auto_save_time 触发器

```
mysql> select * from timelog//
+----+---------------------+
| id | savetime            |
+----+---------------------+
|  1 | 2012-11-23 08:42:28 |
+----+---------------------+
1 row in set (0.00 sec)

mysql>
```

图 17.2　查看 timelog 表中是否执行插入操作

17.1.2　创建具有多个执行语句的触发器

上面 17.1.1 节中，已经介绍了如何创建一个最基本的触发器，但是在实际应用中，往往触发器中包含多个执行语句。其中创建具有多个执行语句的触发器语法结构如下：

```
CREATE TRIGGER 触发器名称 BEFORE | AFTER 触发事件
ON 表名 FOR EACH ROW
BEGIN
执行语句列表
END
```

其中，创建具有多个执行语句触发器的语法结构与创建触发器的一般语法结构大体相同，其参数说明请参考 17.1.1 节中的参数说明，这里不再赘述了。在该结构中，将要执行的多条语句放入 BEGIN 与 END 之间。多条语句需要执行的内容，需要用分隔符“;”隔开。

技巧 一般放在 BEGIN 与 END 之间的多条执行语句必须用结束分隔符“;”分开。在创建触发器过程中需要更改分隔符，故这里应用上一章提到的 DELIMITERT 语句，将结束符号变为“//”。当触发器创建完成后，读者同样可以应用该语句将结束符换回“;”。

下面创建一个由 DELETE 触发多个执行语句的触发器 delete_time_info。模拟一个删除日志数据表和一个删除时间表。当用户删除数据库中的某条记录后，数据库系统会自动向日志表中写入日志信息。

例 17.2　在上例中创建的 timelog 数据表基础上，另外创建一个名称为 timeinfo 的数据表。（**实例位置：光盘\TM\Instance\17\17.2**）

创建代码如下：

```
create table timeinfo(
id int(11), primary key auto_increment,
info varchar(50) not null
)//
```

然后创建一个由 DELETE 触发多个执行语句的触发器 delete_time_info。其代码如下：

```
delimiter //
create trigger delete_time_info after delete
on studentinfo for each row
begin
insert into timelog(savetime) values (now());
insert into timeinfo(info) values ('deleteact');
end
//
```

运行以上代码的结果如图 17.3 所示。

```
mysql> delimiter //
mysql> create trigger delete_time_info after delete
    -> on studentinfo for each row
    -> begin
    -> insert into timelog(savetime) values (now());
    -> insert into timeinfo(info) values ('deleteact');
    -> end
    -> //
Query OK, 0 rows affected (0.05 sec)
```

图 17.3　创建具有多个语句的触发器 delete_time_info

例 17.3　触发器创建成功，当执行删除操作后，timelog 与 timeinfo 表中将会插入两条相关记录。执行删除操作的代码如下：（**实例位置：光盘\TM\Instance\17\17.3**）

```
DELETE FROM studentinfo where sid=7;
```

删除成功后，应用 SELECT 语句分别查看 timelog 数据表与 timeinfo 数据表。其运行结果如图 17.4 和图 17.5 所示。



图 17.4　查看 timelog 数据表信息



图 17.5　查看 timeinfo 数据表信息

从图 17.4 和图 17.5 中可以看出，触发器创建成功后，当用户对 students 表执行 DELETE 操作时，students 数据库中的 timelog 数据表和 timeinfo 数据表中分别被插入操作时间和操作信息。

技巧 在 MySQL 中，一个表在相同的时间和相同的触发时间只能创建一个触发器，如触发时间 INSERT 和触发时间 AFTER 的触发器只能有一个。但是可以定义 BEFORE 的触发器。

17.2　查看触发器

视频讲解：光盘\TM\Video\第 17 章\查看触发器.exe

查看触发器是指查看数据库中已存在的触发器的定义、状态和语法等信息。查看触发器应用 SHOW TRIGGERS 语句。

17.2.1　SHOW TRIGGERS

在 MySQL 中，可以执行 SHOW TRIGGERS 语句查看触发器的基本信息。其基本形式如下：

```
SHOW TRIGGERS;
```

进入 MySQL 数据库，选择 students 数据库并查看该数据库中存在的触发器，其运行结果如图 17.6 所示。

在命令提示符中输入 SHOW TRIGGERS 语句即可查看选择数据库中的所有触发器，但是，应用该查看语句存在一定弊端，即只能查询所有触发器的内容，并不能指定查看某个触发器的信息。这样一来，就会在用户查找指定触发器信息时带来极大不便。故推荐读者只在触发器数量较少的情况下应用 SHOW TRIGGERS 语句查询触发器基本信息。

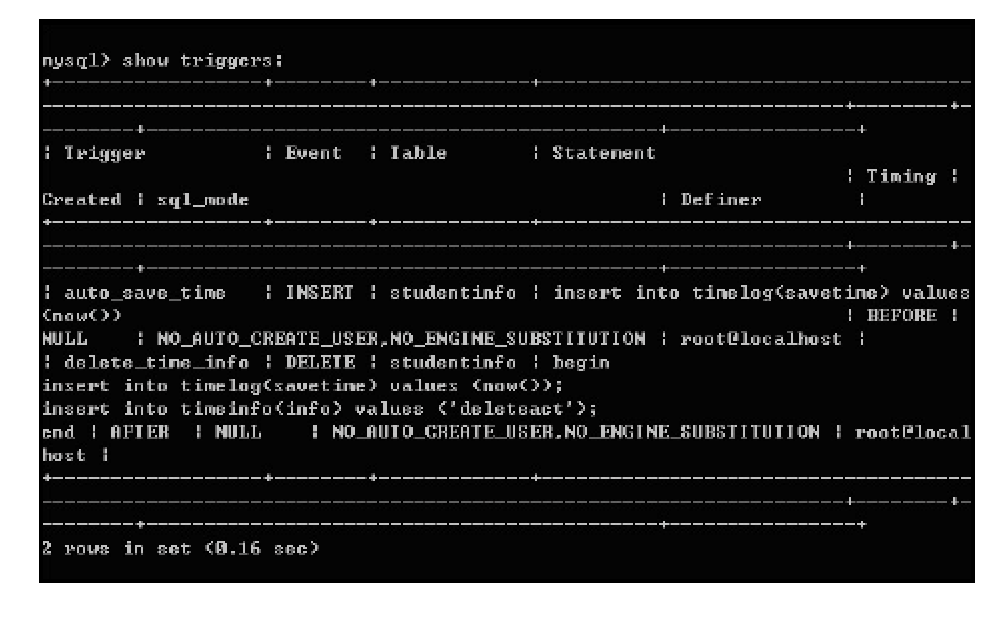


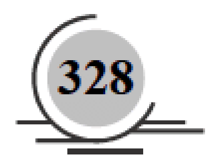
图 17.6　查看触发器

17.2.2　查看 triggers 表中触发器信息

在 MySQL 中，所有触发器的定义都存在该数据库的 triggers 表中。读者可以通过查询 triggers 表来查看数据库中所有触发器的详细信息。查询语句如下：

```
SELECT * FROM information_schema.triggers;
```

其中，information_schema 是 MySQL 中默认存在的库，而 information_schema 是数据库中用于记录触发



器信息的数据表。通过 SELECT 语句查看触发器信息。其运行结果与图 17.6 相同。但是如果用户想要查看某个指定触发器的内容。可以通过 where 子句应用 TRIGGER 字段做为查询条件。其代码如下所示：

```
SELECT * FROM information_schema.triggers WHERE TRIGGER_NAME= '触发器名称';
```

其中，触发器名称这一参数为用户指定要查看的触发器名称，和其他 SELECT 查询语句相同，该名称内容需要用一对“''”（单引号）引用指定的文字内容。

技巧 如果数据库中存在数量较多的触发器，建议读者使用第二种查看触发器的方式。这样会在查找指定触发器过程中避免很多不必要的麻烦。

17.3　应用触发器

视频讲解：光盘\TM\Video\第 17 章\应用触发器.exe

在 MySQL 中，触发器按以下顺序执行：BEFORE 触发器、表操作、AFTER 触发器操作，其中表操作包括常用的数据库操作命令，如 INSERT、UPDATE、DELETE。

例 17.4　触发器与表操作存在执行顺序，下面通过创建一个实例向读者展示 3 者的执行顺序关系。**（实例位置：光盘\TM\Instance\17\17.4）**

（1）创建名称为 before_in 的 BEFORE INSERT 触发器。其代码如下：

```
create trigger before_in before insert on
studentinfo for each row
insert into timeinfo (info) values ('before');
```

（2）创建名称为 after_in 的 AFTER INSERT 触发器。其代码如下：

```
create trigger after_in after insert on
studentinfo for each row
insert into timeinfo (info) values ('after');
```

运行步骤（1）、（2）的结果如图 17.7 所示。

（3）创建完毕触发器，向数据表 studentinfo 中插入一条记录。代码如下：

```
insert into studentinfo(name) values ('Nowitzki');
```

（4）执行成功后，通过 SELECT 语句查看 timeinfo 数据表的插入情况。代码如下：

```
select * from timeinfo;
```

运行以上代码，其运行结果如图 17.8 所示。

```
mysql> create trigger before_in before insert on
    -> studentinfo for each row
    -> insert into timeinfo (info) values ('before');
Query OK, 0 rows affected (0.02 sec)

mysql> create trigger after_in after insert on
    -> studentinfo for each row
    -> insert into timeinfo (info) values ('after');
Query OK, 0 rows affected (0.00 sec)
```

图 17.7　创建触发器运行结果



图 17.8　查看 timeinfo 表中触发器的执行顺序

查询结果显示 before 和 after 触发器被激活。before 触发器首先被激活，然后 after 触发器再被激活。

技巧 触发器中不能包含 START TRANSCATION、COMMIT 或 ROLLBACK 等关键字，也不能包含 CALL 语句。触发器执行非常严密，每一环都息息相关，任何错误都可能导致程序无法向下执行。已经更新过的数据表是不能回滚的。故在设计过程中一定要注意触发器的逻辑严密性。

17.4 删除触发器

视频讲解：光盘\TM\Video\第 17 章\删除触发器.exe

在 MySQL 中，既然可以创建触发器，同样也可以通过命令删除触发器。删除触发器指删除原来已经在某个数据库中创建的触发器，与 MySQL 中删除数据库的命令相似。删除触发器应用 DROP 关键字。其语法格式如下：

```
DROP TRIGGER 触发器名称
```

触发器名称参数为用户指定要删除的触发器名称，如果指定某个特定触发器名称，MySQL 在执行过程中将会在当前库中查找触发器。

技巧 *在应用完触发器后，切记一定要将触发器删除，否则在执行某些数据库操作时，会造成数据的变化。*

例 17.5 下面将名称为 delete_time_info 的触发器删除。（**实例位置：光盘\TM\Instance\17\17.5**）

其执行代码如下：

```
DROP TRIGGER delete_time_info;
```

执行上述代码，其运行结果如图 17.9 所示。

通过查看触发器命令来查看数据库 students 中的触发器信息。其代码如下：

```
SHOW TRIGGERS
```

查看触发器信息，可以从图 17.10 看出，名称为 delete_time_info 的触发器已经被删除了。

```
mysql> use students;
Database changed
mysql> drop trigger delete_time_info;
Query OK, 0 rows affected (0.11 sec)
```

图 17.9 删除触发器

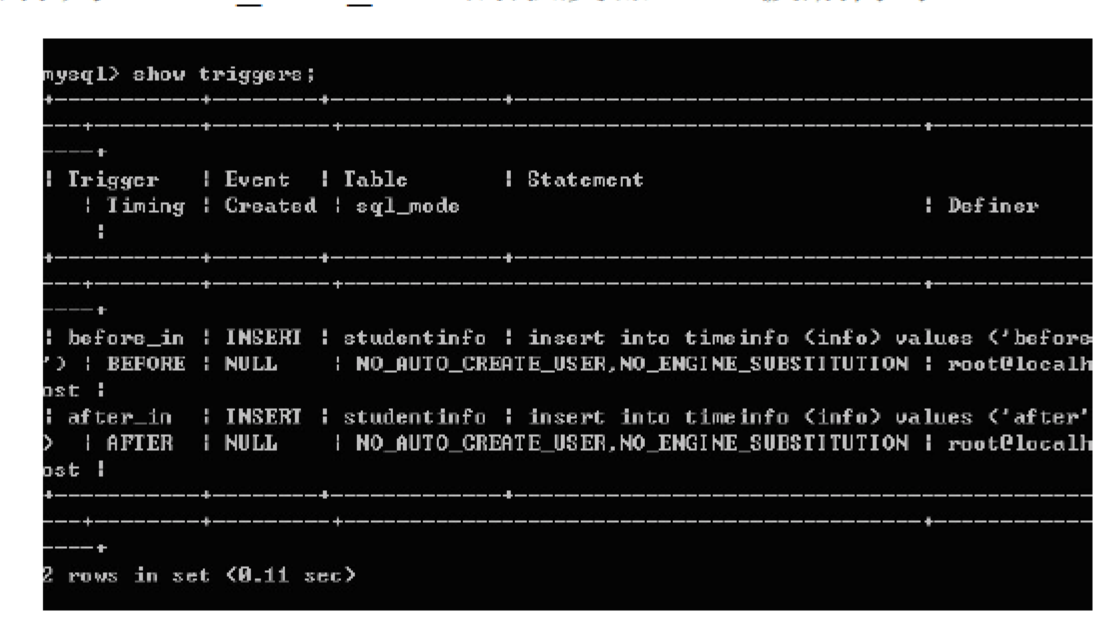


图 17.10 查看 students 数据库中的触发器信息

注意 *图 17.10 的返回结果显示，该数据库中存在两个触发器信息，这两个触发器是在 17.1.2 节中被创建的，如果用户在 db_database17 数据库中未创建该触发器，则返回结果会是一个"Empty set"。*

17.5 实　　战

17.5.1 创建一个由 INSERT 触发的触发器

例 17.6 下面向 department 表中插入一条记录，然后查看 tb_students 表中是否执行 insert 操作。（**实例**



位置：光盘\TM\Instance\17\17.6）

代码如下：

```
CREATE TRIGGER trig1 BEFORE INSERT
ON department FOR EACH ROW
INSERT INTO tb_students(times) VALUES(NOW());
INSERT INO department(name) values('liliy');
Select * from tb_students;
```

执行上述代码，其运行结果如图 17.11 所示。

17.5.2 获取数据库中的触发器

例 17.7 在图 17.12 中的文本框中输入要查看数据库和触发器，然后单击“查看”按钮即可将该数据库中所有的触发器的详细信息显示出来。效果如图 17.12 所示。（**实例位置：光盘\TM\Instance\17\17.7**）

```
mysql> create trigger trig1 before insert
    -> on department for each row
    -> insert into tb_students(times) values(now());
Query OK, 0 rows affected (0.05 sec)

mysql> insert into department(name) values('lily');
Query OK, 1 row affected (0.00 sec)

mysql> select * from tb_students;
+----+---------------------+
| id | times               |
+----+---------------------+
|  1 | 2012-07-24 08:34:56 |
|  2 | 2012-07-24 09:08:27 |
+----+---------------------+
2 rows in set (0.00 sec)

mysql> _
```

图 17.11 创建一个由 INSERT 触发的触发器



名称	涉及表名	触发事件	时间	触发时执行的SQL语句
trig1	department	INSERT	BEFORE	INSERT INTO tb_students(times) VALUES(NOW())
auto_save_time	studentinfo	INSERT	BEFORE	insert into timelog(savetime) values(now())
after_in	studentinfo	INSERT	AFTER	insert into timeinfo (info) values ('after')

图 17.12 获取数据库中的触发器

实现过程如下。

（1）选择要查看数据库的触发器。代码如下：

```
  <?php
          $dbname=$_POST['name'];
          $conn=mysql_connect("localhost","root","111");
        mysql_select_db($dbname,$conn);
        mysql_query("set names gb2312");
```

（2）执行 show trigger 语句并显示所查找到的触发器的详细信息。代码如下：

```
<?php
$sql=@mysql_query("show triggers",$conn);
      $info=@mysql_fetch_array($sql);
      if($info==false){
      ?>
      <tr>
     <td height="25" colspan="5" bgcolor="#FFFFFF"><div align="center">该数据库没有设置触发器!</div></td>
</tr>
      <?php
      }else{
           do{
      ?>
    <tr>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info['Trigger'];?></div></td>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info['Table'];?></div></td>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info['Event'];?></div></td>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info['Timing'];?></div></td>
          <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info['Statement'];?></div></td>
        </tr>
```

```
	<?php
		}while($info=mysql_fetch_array($sql));
}
?>
```

17.5.3　使用 DROP TIRGGER 删除触发器

例 17.8　删除触发器指删除原来已经在某个数据库中创建的触发器，与 MySQL 中删除数据库的命令相似。删除触发器应用 DROP 关键字。**（实例位置：光盘\TM\Instance\17\17.8）**

代码如下：

```
DROP TRIGGER trig1
SHOW TRIGGERS;
```

执行上述代码，删除前后的效果如图 17.13 和图 17.14 所示。

图 17.13　删除触发器之前

图 17.14　删除触发器之后

17.6　小　　结

本章对 MySQL 数据库的触发器的定义和作用、创建触发器、查看触发器、使用触发器和删除触发器等内容进行了详细讲解，创建触发器和使用触发器是本章的重点内容。读者在创建触发器后，一定要查看触发器的结构。使用触发器时，触发器执行的顺序为 BEFORE 触发器、表操作（INSERT、UPDATE 和 DELETE）和 AFTER 触发器。读者需要将本章的知识结合实际需要来设计触发器。

17.7　学习成果检验

1．如何查看触发器信息？**（实例位置：光盘\TM\Instance\17\17.9）**

2．触发器的触发顺序是什么？**（实例位置：光盘\TM\Instance\17\17.10）**

3．触发器执行语句的限制条件是什么？**（实例位置：光盘\TM\Instance\17\17.11）**

4．MySQL 中创建多条执行语句的触发器总是遇到分号就结束创建，然后报错，为什么？**（实例位置：光盘\TM\Instance\17\17.12）**

第18章

综合实例（三）——物流管理系统

（视频讲解：83分钟）

物流信息化是指物流企业运用现代信息技术对物流过程中产生的全部或部分信息进行采集、分类、传递、汇总、查询等一系列处理活动，以实现对货物流动过程的控制，从而降低成本、提高效益的管理活动。

物流企业信息化的目的就是要满足企业自身管理的需要和不同类型企业在物流业务外包过程中对信息交换的要求，也就是通过建设物流信息系统，提高信息流转效率，降低物流运作成本。

目前我国正处于全面推进信息化的进程中，所以物流领域的信息化已成为一个必然。物流信息化将成为现代物流的灵魂，将是现代物流发展的必经之路。

通过阅读本章内容，你可以：

- 了解物流配送系统开发的基本过程
- 熟悉如何创建 MySQL 数据库的存储过程
- 熟悉如何通过 MySQL 数据库存储过程实现用户登录
- 熟悉如何实现数据库中数据的模糊查询技术
- 熟悉如何创建浮动框架
- 熟悉如何实现订单的打印技术

18.1　物流管理系统概述

随着我国信息化进程的全面推进，各领域的信息化进程都在飞速发展，同样也推动着物流领域的信息化进程飞快地向前发展。由于信息化进程的全面推进，对现代物流提出更高的要求：信息化、自动化、网络化、智能化和人性化等。物流行业的竞争日益激烈，客户需求的标准也越来越高，物流企业要想在市场中占有一席之地，必须要建立一个高效、快捷、方便的物流信息系统，为客户提供一流的服务，并且能够及时准确地掌握客户的需求，对客户的需求做出快速反映，在最短的时间内以最大限度挖掘和优化物流资源来满足客户的需求，从而建立高效的数字化物流经济。

18.2　系统分析流程

18.2.1　需求分析

随着现代物流信息化进程的加快，传统的物流管理方式已经不再适应当前物流发展的要求，取而代之的将是以计算机为基础的网络化物流管理方式。某物流配送公司为适应物流信息化进程的发展，急需开发一个物流配送系统，通过网络来实现对物流操作流程进行管理，不但为企业的运营过程节省大量的人力、物力、财力和时间，提高物流系统运行的效率，而且为企业在客户中树立一个全新的形象，为企业的发展奠定一个良好的基础。现根据对该物流配送公司的实际调查，以及公司的具体要求，制定出物流管理系统的规划方案。具体内容如下：

- ☑ 网站页面设计要求美观大方，能够展示企业形象。
- ☑ 为企业在客户中树立一个全新的形象。
- ☑ 网站的操作流程简单、方便，能够提高工作效率。
- ☑ 提供物流配送的全程跟踪。
- ☑ 提供配送信息的及时查询。
- ☑ 实现对配送车辆的管理。
- ☑ 实现对客户信息的管理。
- ☑ 实现发货单打印的功能。

18.2.2　可行性分析

物流管理系统的开发不但能使物流企业走上科学化、网络化管理的道路，而且能够为企业带来巨大的经济效益和技术上飞速的发展。

1．经济性

科学的管理方法，便捷的操作环境，系统的经营模式，将为企业带来更多的客户资源，树立企业的品牌形象，提高企业的经济效益。

2．技术性

网络化的物流管理方式，在操作过程中能够快捷地查找出车源信息、客户订单以及客户信息；能够对



货物进行全程跟踪，了解货物的托运情况，从而使企业能够根据实际情况，做好运营过程中的各项准备工作，作出及时准确的调整；能够保证托运人以及收货人对货物进行及时地处理。

18.3　系统设计流程

18.3.1　系统目标

结合目前网络上物流配送系统的设计方案，对客户做的调查结果以及企业的实际需求。本项目在设计时应该满足以下目标：

- ☑　界面设计美观大方、操作简单。
- ☑　功能完善、结构清晰。
- ☑　能够快速查询车源信息。
- ☑　能够准确填写订单。
- ☑　能够实现订单查询、打印。
- ☑　能够实现对回单处理。
- ☑　能够对车源信息进行添加、修改和删除。
- ☑　能够对客户信息进行管理。
- ☑　能够及时、准确地对网站进行维护和更新。
- ☑　良好的数据库系统支持。
- ☑　系统运行稳定，具备良好的防范措施。

18.3.2　系统功能结构

结合需求分析和系统目标中的内容，物流管理系统的功能结构已经设计完成。为了使读者能够更清楚地了解网站的结构，下面给出物流管理系统的功能模块结构图和工作流程图。

物流管理系统的功能模块结构图，如图 18.1 所示。

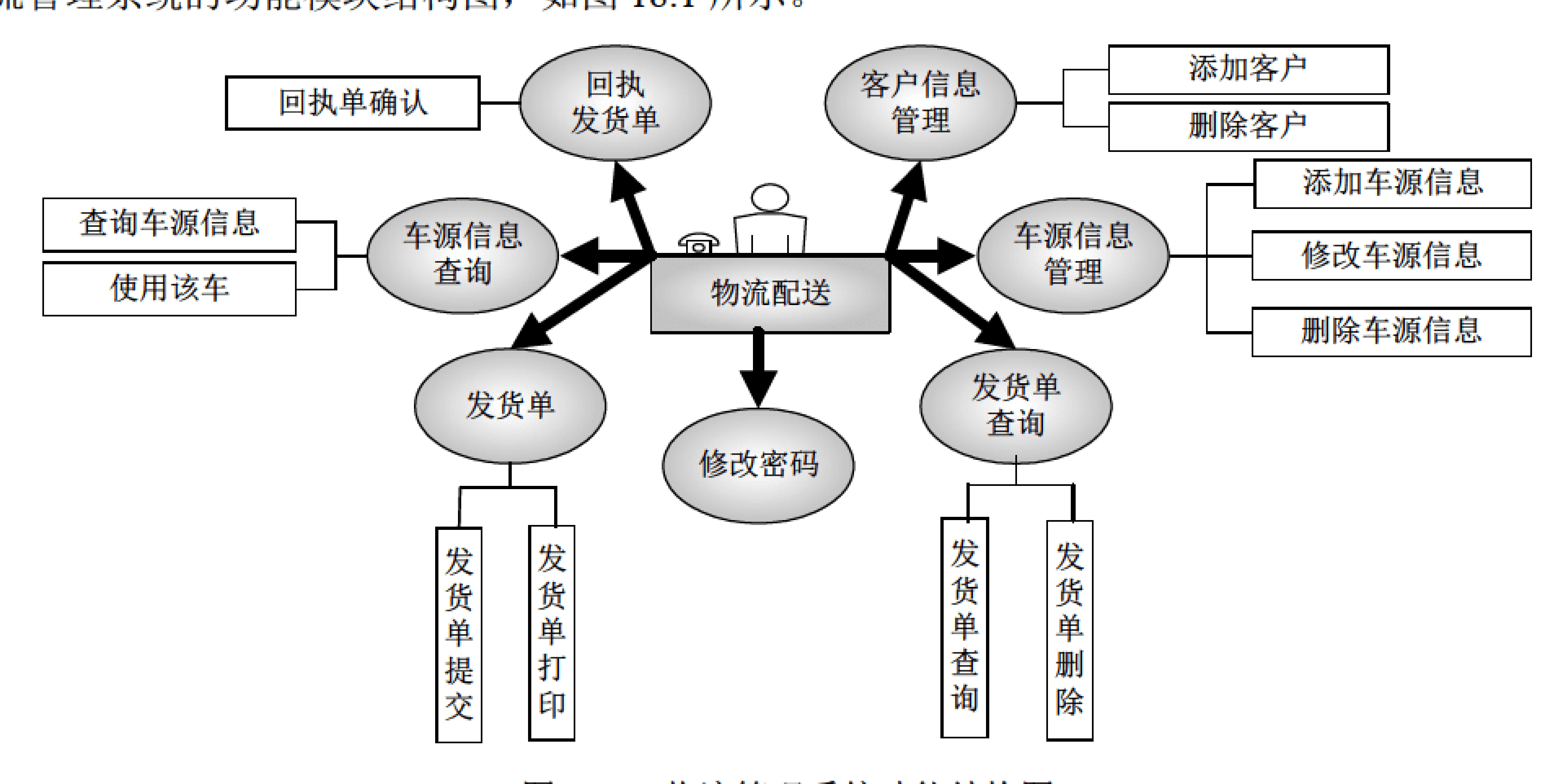


图 18.1　物流管理系统功能结构图

物流管理系统的工作流程图如图 18.2 所示。

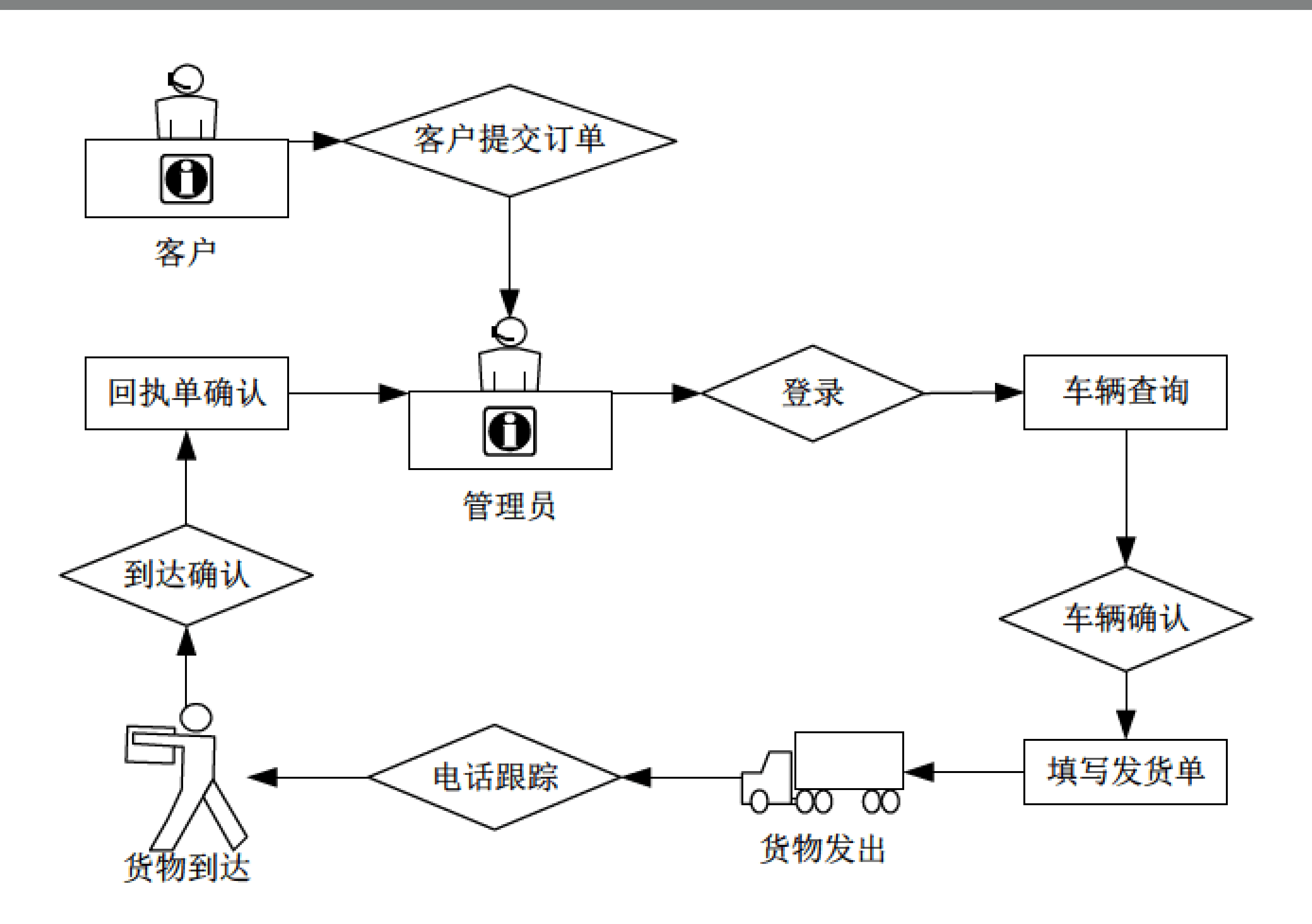


图 18.2　物流管理系统的工作流程图

18.3.3　系统预览

物流管理系统由多个程序页面组成，下面给出几个典型页面，其他页面参见光盘中的源程序。物流管理系统主页如图 18.3 所示，该页面用于展示本系统的车源信息查询模块。登录页面如图 18.4 所示。



图 18.3　主页



图 18.4　登录页面

回执发货单确认模块的页面效果如图 18.5 所示，该页面主要用于实现对回执发货单的处理。发货单查询模块的页面效果如图 18.6 所示，该页面主要用于实现对指定发货单的查询，并且输出发货单的详细内容。



图 18.5　回执发货单



图 18.6　发货单查询

车源信息管理模块的页面效果如图 18.7 所示，该页面实现对车源信息的添加、修改和删除操作。修改管理员密码模块的页面效果如图 18.8 所示，该页面实现对管理员密码的修改。



图 18.7　车源信息管理



图 18.8　修改管理员密码

18.4　数据库设计

视频讲解：光盘\TM\Video\第 18 章\数据库设计.exe

物流管理系统必须拥有数据库的支持，所有物流配送的数据都应该存储到数据库中，便于管理员查找车辆、订单和客户的信息。如果没有数据库的支持，那么物流管理系统将没有任何意义。本节将对物流管理系统的数据库设计进行详细介绍。

18.4.1　数据库分析

物流管理系统是一个中小型的企业管理系统，考虑到开发的成本、搭配的合理性以及操作的灵活性等，使用 MySQL 数据库是最佳的选择。MySQL 数据库是完全免费的，使用它不需要任何费用，可以直接从网上免费下载；MySQL 数据库的操作也非常方便，不但可以在命令模式下操作，而且配备目前比较流行的图形化管理工具 phpMyAdmin，能够轻松地实现对 MySQL 数据库的管理和操作。

18.4.2　数据库概念设计

根据上述各节对物流管理系统做的需求分析和系统设计，整理出物流管理系统的实体关系 E-R 图。其中包括管理员信息实体、车源信息实体、车辆日志信息实体、客户信息实体和发货单信息实体。

1．管理员信息实体

管理员信息实体用于存储管理员的登录名称和密码信息，包括用户名和密码两项内容。管理员信息实体的 E-R 图如图 18.9 所示。

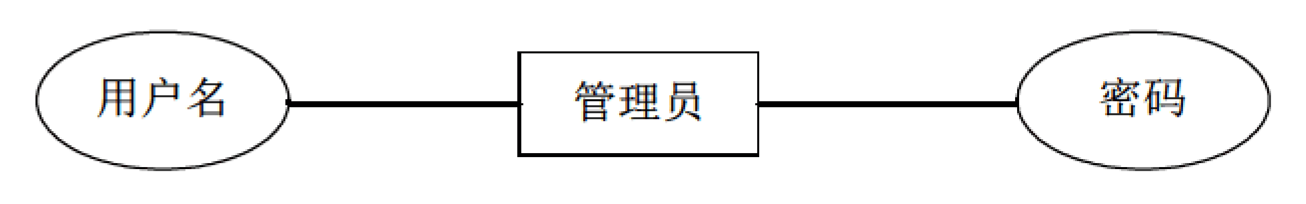


图 18.9　管理员信息实体的 E-R 图

2．车源信息实体

车源信息实体用于存储企业拥有的车辆信息，包括车主姓名、车主身份证号码、车牌号码、车主联系电话、车主的家庭地址、车辆行驶的路线和车辆描述。车源信息实体的 E-R 图如图 18.10 所示。

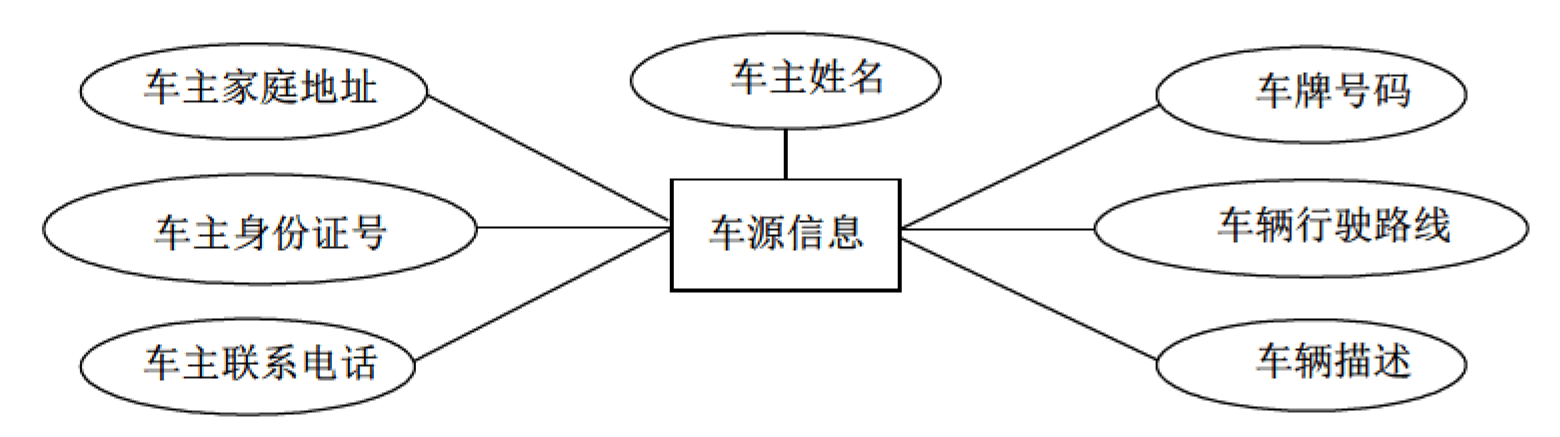


图 18.10　车源信息实体的 E-R 图

3．车辆日志信息实体

车辆日志信息实体记录车辆当前是否被占用，以及使用的时间、执行的任务，包括车牌号码、车辆日志信息（详细的车辆使用描述）、日志时间和发货单 ID（即执行的任务）。车辆日志信息实体的 E-R 图如图 18.11 所示。

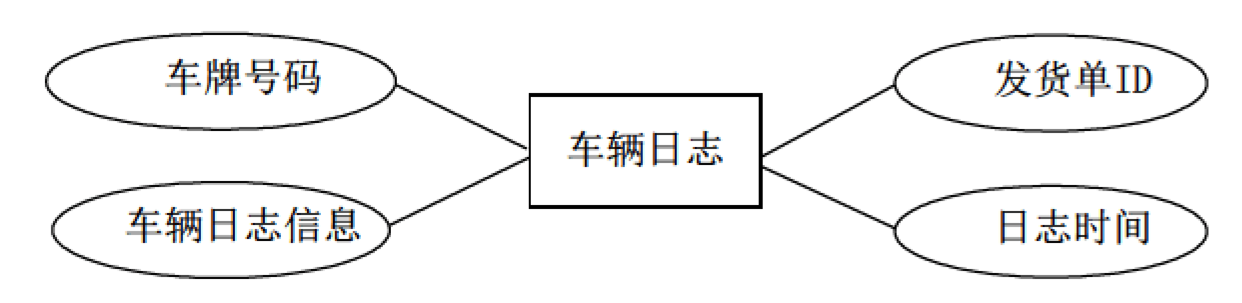


图 18.11　车辆日志信息实体的 E-R 图

4．客户信息实体

客户信息实体用于存储客户的信息，包括客户的姓名、客户电话和客户的联系地址。客户信息实体的 E-R 图如图 18.12 所示。

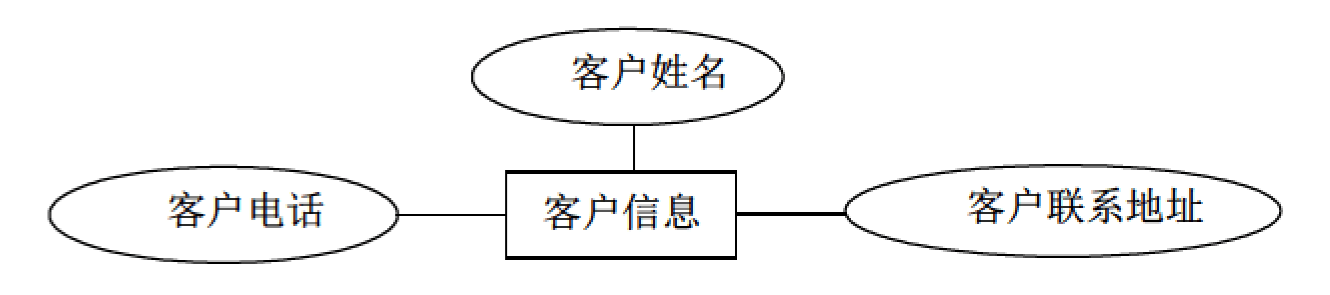


图 18.12　客户信息实体 E-R 图

5．发货单信息实体

发货单信息实体用于存储客户填写的发货单信息，包括车牌号码、车主电话、货物描述、发货人、发货时间、发货人电话、发货地址、收货人、收货人电话、收货地址和付款方式。发货单信息实体的 E-R 图如图 18.13 所示。

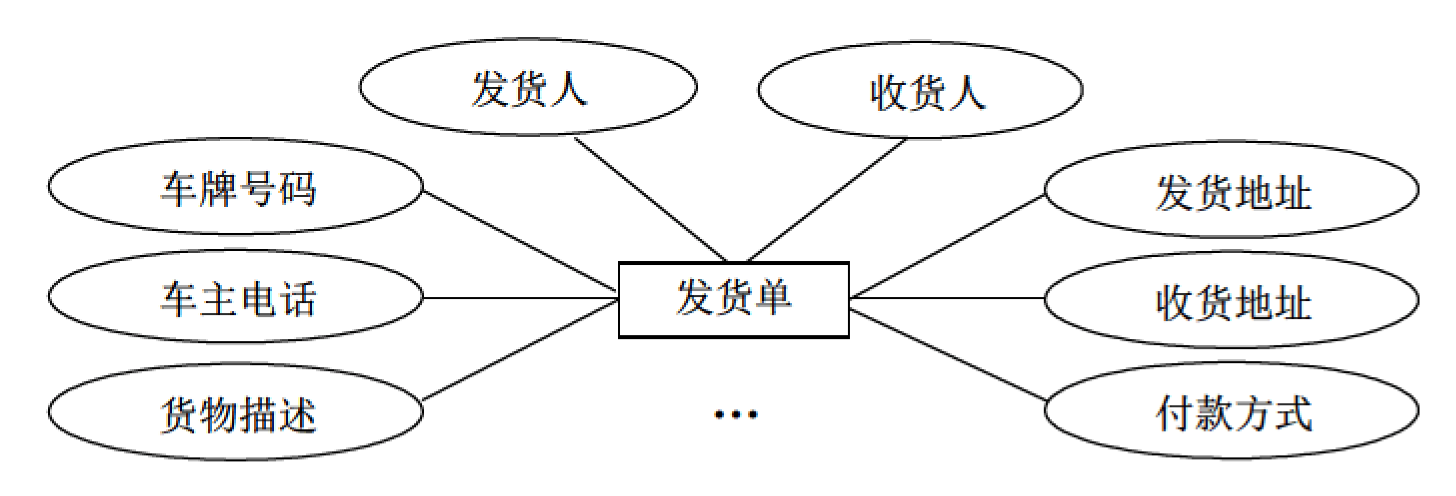
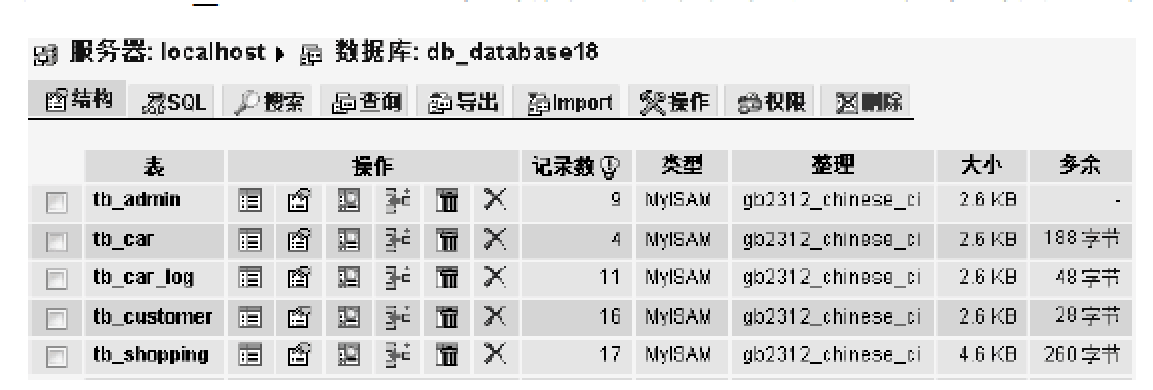


图 18.13　发货单信息实体的 E-R 图

18.4.3　创建数据库及数据表

在物流管理系统中应用的是 db_database18 数据库，其中涉及 5 个数据表，如图 18.14 所示。


服务器: localhost ▸ 数据库: db_database18

结构 SQL 搜索 查询 导出 Import 操作 权限 删除

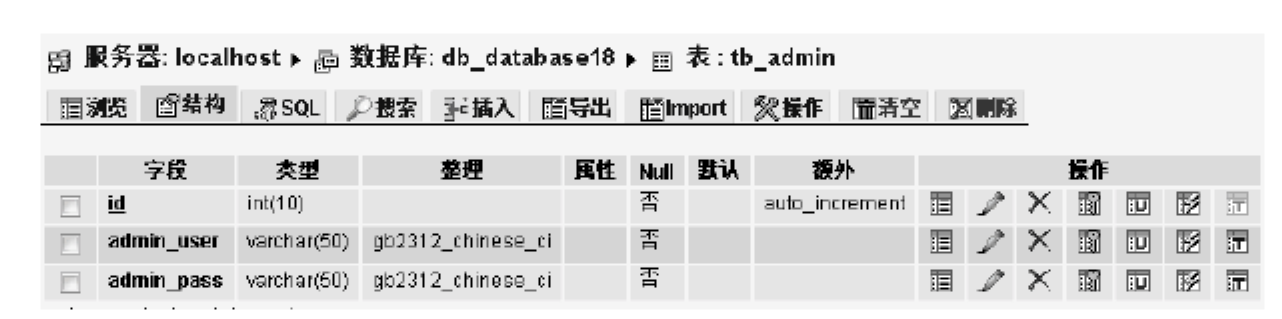
表	操作	记录数	类型	整理	大小	多余
tb_admin		9	MyISAM	gb2312_chinese_ci	2.6 KB	-
tb_car		4	MyISAM	gb2312_chinese_ci	2.6 KB	188 字节
tb_car_log		11	MyISAM	gb2312_chinese_ci	2.6 KB	48 字节
tb_customer		16	MyISAM	gb2312_chinese_ci	2.6 KB	28 字节
tb_shopping		17	MyISAM	gb2312_chinese_ci	4.6 KB	260 字节

图 18.14　物流管理系统的数据库

下面介绍物流管理系统中使用的数据表，数据表的设计结构如图 18.15～图 18.19 所示。

1．tb_admin（管理员信息表）

管理员信息表主要用于存储管理员的登录名和密码。该数据表的结构如图 18.15 所示。


服务器: localhost ▸ 数据库: db_database18 ▸ 表 : tb_admin

浏览 结构 SQL 搜索 插入 导出 Import 操作 清空 删除

字段	类型	整理	属性	Null	默认	额外	操作
id	int(10)			否		auto_increment	
admin_user	varchar(50)	gb2312_chinese_ci		否			
admin_pass	varchar(50)	gb2312_chinese_ci		否			

图 18.15　管理员信息表结构

2．tb_car（车源信息表）

车源信息表主要用于存储配送公司拥有的车辆信息，主要包括车主姓名、车主身份证号码、车牌号码、车主联系电话、车主的家庭地址、车辆行驶的路线和车辆描述。该数据表的结构如图 18.16 所示。


服务器: localhost ▸ 数据库: db_database18 ▸ 表 : tb_car

浏览 结构 SQL 搜索 插入 导出 Import 操作 清空 删除

字段	类型	整理	属性	Null	默认	额外	操作
id	int(10)			否		auto_increment	
username	varchar(50)	gb2312_chinese_ci		否			
user_number	varchar(50)	gb2312_chinese_ci		否			
car_number	varchar(50)	gb2312_chinese_ci		否			
tel	varchar(50)	gb2312_chinese_ci		否			
address	varchar(80)	gb2312_chinese_ci		否			
car_road	mediumtext	gb2312_chinese_ci		否			
car_content	mediumtext	gb2312_chinese_ci		否			

图 18.16　车源信息表结构

3．tb_car_log（车辆日志信息表）

车辆日志信息表主要用于存储车辆的使用情况信息，主要包括车牌号码、车辆使用日志、日志创建时间和发货单 ID。该数据表的结构如图 18.17 所示。


服务器: localhost ▸ 数据库: db_database18 ▸ 表 : tb_car_log

浏览 结构 SQL 搜索 插入 导出 Import 操作 清空 删除

字段	类型	整理	属性	Null	默认	额外	操作
log_id	int(10)			否		auto_increment	
car_number	varchar(50)	gb2312_chinese_ci		否			
car_log	mediumtext	gb2312_chinese_ci		否			
log_date	datetime			否			
fahuo_id	varchar(50)	gb2312_chinese_ci		否			

图 18.17　车辆日志信息表结构

4．tb_customer（客户信息表）

客户信息表主要用于存储客户的信息，包括姓名、电话以及联系地址。该数据表的结构如图 18.18 所示。

服务器: localhost ▸ 数据库: db_database18 ▸ 表 : tb_customer

浏览 结构 SQL 搜索 插入 导出 Import 操作 清空 删除

字段	类型	整理	属性	Null	默认	额外	操作
customer_id	int(10)			否		auto_increment	
customer_user	varchar(50)	gb2312_chinese_ci		否			
customer_tel	varchar(50)	gb2312_chinese_ci		否			
customer_address	varchar(80)	gb2312_chinese_ci		否			

图 18.18　客户信息表结构

5．tb_shopping（发货单信息表）

发货单信息表主要用于存储客户填写的发货单中的信息，主要包括车牌号码、车主电话、货物描述、发货人、发货时间、发货人电话、发货地址、收货人、收货人电话、收货地址和付款方式。该数据表的结构如图 18.19 所示。

服务器: localhost ▸ 数据库: db_database18 ▸ 表 : tb_shopping

浏览 结构 SQL 搜索 插入 导出 Import 操作 清空 删除

字段	类型	整理	属性	Null	默认	额外	操作
id	int(10)			否		auto_increment	
car_number	varchar(50)	gb2312_chinese_ci		否			
fahuo_content	mediumtext	gb2312_chinese_ci		否			
fahuo_id	varchar(50)	gb2312_chinese_ci		否			
fahuo_user	varchar(50)	gb2312_chinese_ci		否			
fahuo_time	datetime			否			
fahuo_ys	varchar(20)	gb2312_chinese_ci		是	0		
fahuo_fk	varchar(20)	gb2312_chinese_ci		否			
car_tel	varchar(50)	gb2312_chinese_ci		否			
shouhuo_user	varchar(50)	gb2312_chinese_ci		否			
shouhuo_address	mediumtext	gb2312_chinese_ci		否			
fahuo_address	mediumtext	gb2312_chinese_ci		否			
fahuo_tel	varchar(50)	gb2312_chinese_ci		否			
shouhuo_tel	varchar(50)	gb2312_chinese_ci		否			

图 18.19　发货单信息表结构

18.5　网站首页设计

视频讲解：光盘\TM\Video\第 18 章\网站首页设计.exe

18.5.1　网站首页概述

网站首页是整个网站的门面，是浏览者看到的第一视觉界面，所以在设计网站的首页时应该将网站中主要的内容尽量展示给浏览者，让浏览者能够更快地了解网站的内容。物流管理系统的首页主要包括如下功能。

☑　车源信息查询：主要实现车辆信息查询，及时为客户选择货物配送的方案。

☑　发货单：主要用于客户发货单信息填写。

☑　回执发货单确认：主要用于对货物配送回执单的确认。

☑　发货单查询：主要用于对发货单进行查询。

☑　客户信息管理：主要用于对客户的信息进行管理。

☑　车源信息管理：主要用于对企业拥有的车辆信息进行管理。

☑　修改密码：主要用于对管理员登录的密码进行修改。

下面看一下本案例中提供的网站首页，该页面在本书光盘中的路径为\TM\18\index.php，如图 18.20 所示。

18.5.2　网站首页设计技术

物流管理系统是为物流配送公司设计的一个物流管理系统，其页面的设计突出简洁、方便，功能的实现以便于操作和维护为根本。网站首页的页面设计如图 18.20 所示，由一个简单的框架组成，其结构由两部分组成：一部分用于输出网站中包括的模块，另一部分用于输出对应模块中的内容。首页框架结构如图 18.21 所示。

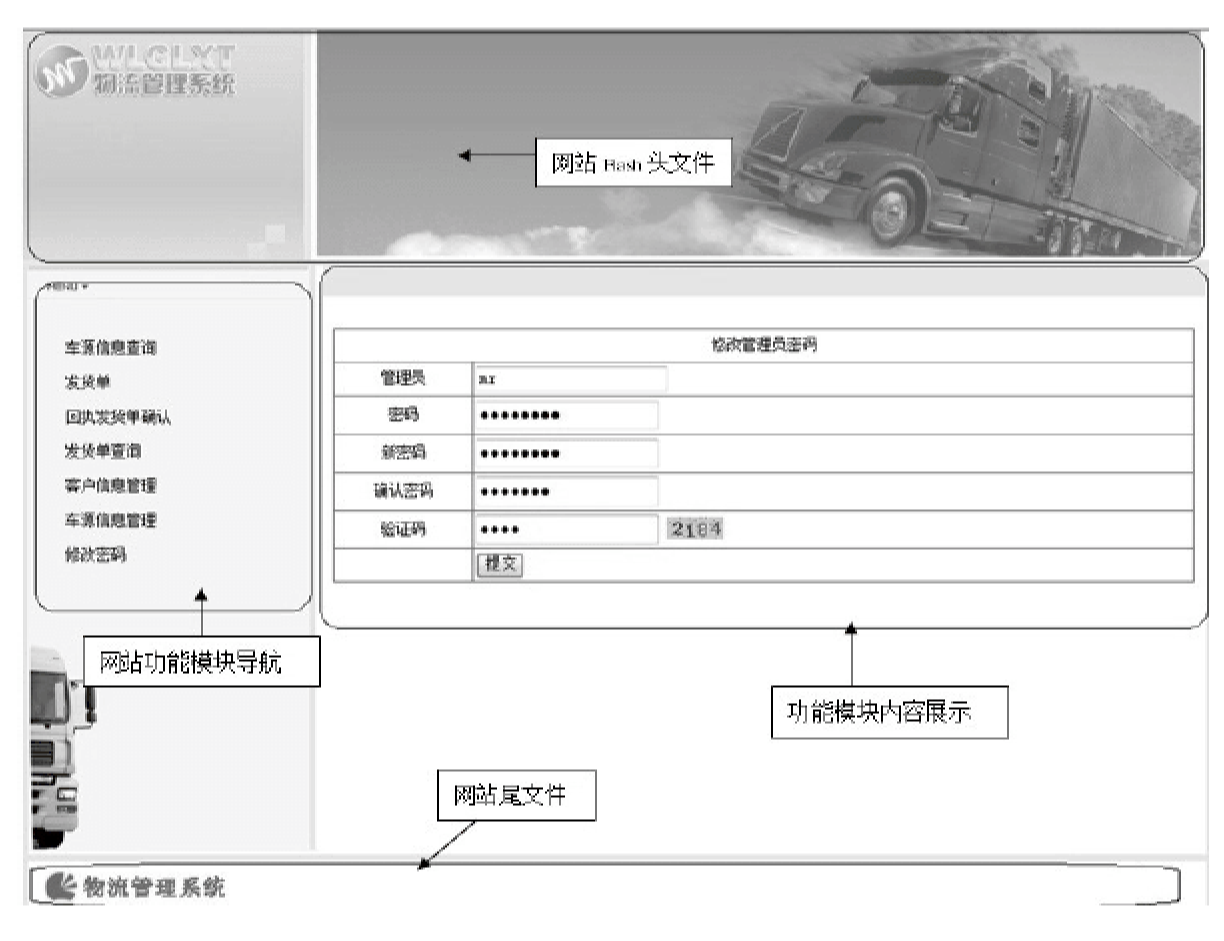


图 18.20　物流管理系统首页

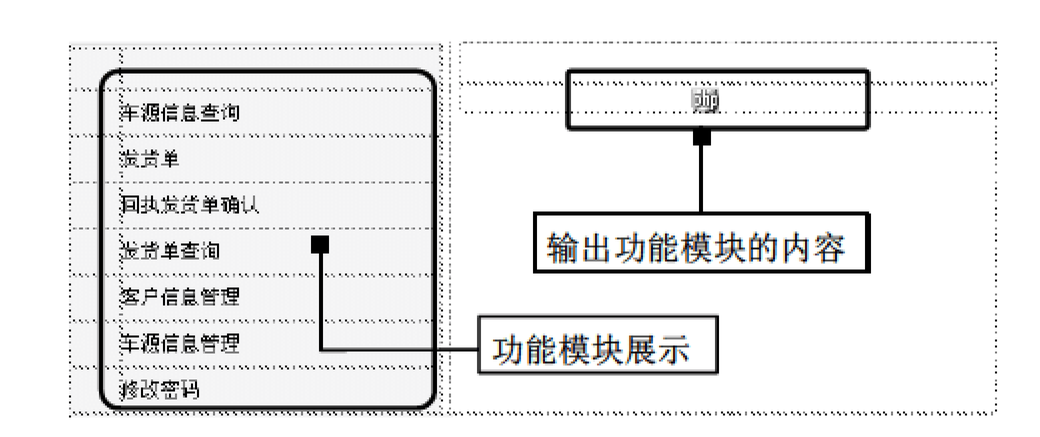


图 18.21　首页框架结构

这里使用的框架结构不是 frame 框架，而是应用 switch 语句，加上表格的嵌套实现的框架效果。

18.5.3　网站首页的实现过程

物流管理系统的首页设计非常简单，主要应用一个简单的框架功能，并且以超链接作为辅助，最终实现在不同功能模块之间的跳转。

首先连接数据库，判断登录用户的身份，如果是管理员则可以进入网站进行操作，否则将提示“请您正确登录！”，并且跳转到网站的登录页面。其关键代码如下：

（代码位置：光盘\TM\Instance\18\indexs.php）

```
<?php    session_start();                              //初始化 session 变量
include("conn/conn.php");                              //连接数据库
if($_SESSION['admin_user']==true){                     //判断登录用户的身份
?>
<!-->省略了部分代码<!-->
<?php
}else{                                                 //没有以管理员登录，则跳转到登录页
    echo "<script>alert('请您正确登录!'); window.location.href='index.php';</script>";
}
?>
```

然后设计首页中使用的模块超链接，并且为每个超链接定义变量标识，当单击不同功能模块超链接时提交不同的变量标识，作为下一步获取模块中详细内容的依据；其中还使用 ononMouseOver 和 onMouseOut 事件，控制鼠标移动到和离开单元格时显示不同的颜色。部分代码如下：

```
<!-- ----------------------应用 onMouseOver 和 onMouseOut 事件控制鼠标移过和离开时的时间------------------------ -->
<tr onMouseOver="this.bgColor='#9FCB3A'"onMouseOut="this.bgColor='#F5F5F5'" >
```

```
        <td height="28" align="center"> </td>
        <!-- -------------------设置车源信息查询模块的超链接，变量标识为"lmbs=车源信息查询"------------------ -->
       <td align="left"><a href="indexs.php?lmbs=车源信息查询">车源信息查询</a></td>
</tr>
<tr onMouseOver="this.bgColor='#9FCB3A'"onMouseOut="this.bgColor='#F5F5F5'" >
        <td height="28" align="center"> </td>
        <!-- ---------------------------设置发货单模块的超链接，变量标识为"lmbs=发货单"--------------------------- -->
       <td align="left"><a href="indexs.php?lmbs=发货单">发货单</a></td>
</tr>
```

最后应用 switch 语句，根据变量标识$lmbs 提交的值进行判断，使用 include 包含语句调用不同功能模块的脚本文件。其关键代码如下：

```
<?php
        switch($lmbs){                              //获取变量$lmbs 的值
                case "车源信息查询":                 //判断当变量$lmbs 的值是“车源信息查询”时输出下面的内容
                        include("car_select.php");  //调用指定的包含文件
                break;                              //跳出循环
                                                    //这里省略了部分代码
                case "":                            //判断当变量的值为空时
                        include("car_select.php");  //当变量的值为空时则调用该文件
                break;                              //跳出循环
        }
?>
```

18.6　车源信息查询模块设计

视频讲解：光盘\TM\Video\第 18 章\车源信息查询模块设计.exe

18.6.1　车源信息查询模块概述

车源信息查询模块的功能是根据客户提供的货物配送的地点，从数据库中查询有关该路线的车辆，为客户提供一个合理配送的路线，最后确定发货订单。操作流程如图 18.22 所示。

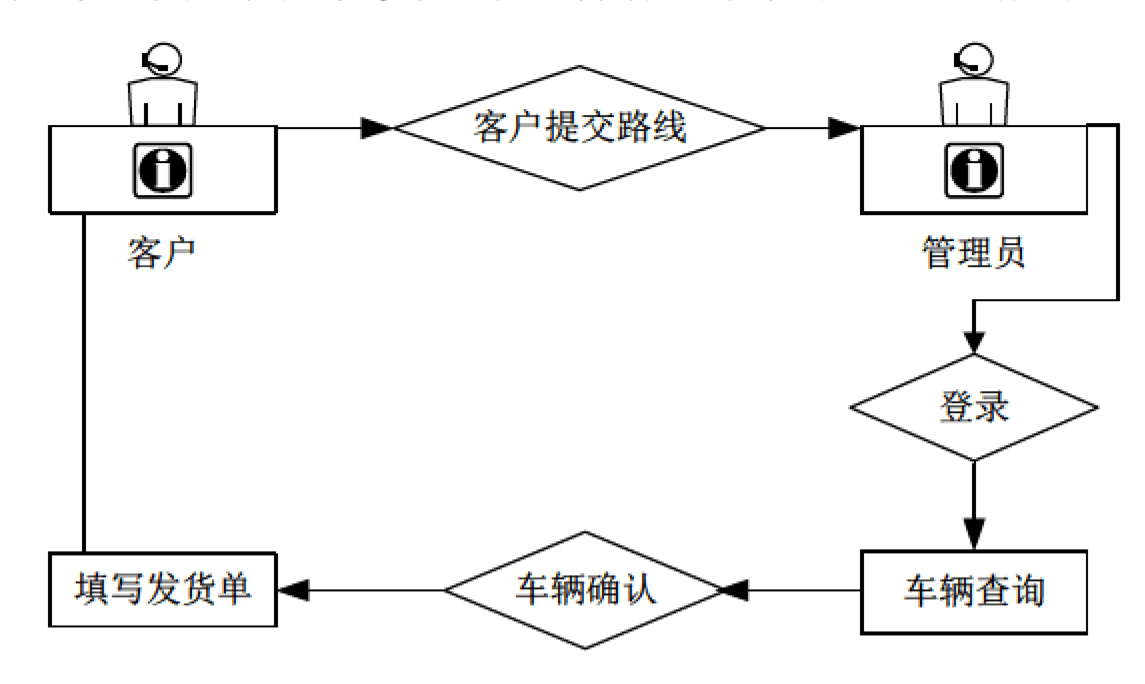


图 18.22　车源信息查询的操作流程

18.6.2　模糊查询技术

车源信息查询模块的主要功能就是查询车辆的使用信息，为客户提供最合适的货物配送路线。其中应



用的关键技术自然就是查询方法，为了给客户提供最合适、最满意、最快捷的配送服务，这里使用的是模糊查询技术。

通过模糊查询技术，只要客户提出配送货物的出发地点和到达地点，管理员就可以从数据库中提取出所有与该路线相关的车辆信息，包括车辆的承载能力、车辆是否被占用和车辆的使用情况，都一目了然，客户可以根据实际的情况（时间、货物数量、路线等）选择车辆，确认后填写发货单。

模糊查询技术的实现是在 car_select.php 文件中完成的，首先要与数据库建立连接；然后创建一个 form 表单，设置两个文本框 select1 和 select2 用于提交出发地点和到达地点，将表单的值提交到当前页；最后编写 PHP 脚本语句，以表单中获取的数据为条件，执行 like 模糊查询，从数据库的指定表中查询符合条件的信息，并且将查询出的信息显示到页面中。程序的关键代码如下：

（代码位置：光盘\TM\Instance\18\car_select.php）

```
<?php
 session_start(); include("conn/conn.php");                        //连接数据库
 ?>
<!-- -------------------------------------创建 form 表单，提交出发地点和到达地点----------------------------------------- -->
<form id="form1" name="form1" method="post" action="indexs.php?lmbs=车源信息查询" onSubmit="javascript:
return check_select_car();">
    <td height="30" colspan="4" align="right" bgcolor="#FFFFFF">查询
❶          <input name="select1" type="text" id="select1" size="10">至
❷          <input name="select2" type="text" id="select2" size="10"></td>
    <td colspan="2" bgcolor="#FFFFFF"><input type="submit" name="Submit" value="提交" /></td>
    </form>
<!-- --------------------获取表单提交的值，根据该条件执行 like 模糊查询，从数据库中读取数据-------------------- -->
<?php
❸    if($_POST['select1']==true || $_POST['select2']==true){     //判断表单提交的值是否为真
     /* *******************以表单提交的值为条件，从数据表中查询数据***************** */
❹    $query="select * from tb_car where car_road like '%$select1%' and car_road like '%$select2%'";
     $result=mysql_query($query);                               //执行查询语句
     if(mysql_num_rows($result)>0){                             //判断是否存在符合条件的数据
     while($myrow=mysql_fetch_array($result)){                  //应用 while 循环语句，输出符合条件的数据
?>
<tr>
     <td height="26" align="center" bgcolor="#FFFFFF"><?php echo $myrow['car_number'];?></td>
    <td bgcolor="#FFFFFF"><?php echo $myrow['car_road'];?></td>
    <td colspan="2" bgcolor="#FFFFFF"><?php echo $myrow['car_content'];?></td>
    <td bgcolor="#FFFFFF">
     <?php
          /* **********根据车牌号码，从车辆日志信息表中读取对应车辆的使用信息********** */
          $querys=mysql_query("select * from tb_car_log where car_number='$myrow['car_number']'");
          $myrows=mysql_fetch_array($querys);
          echo $myrows['car_log'];                              //输出车辆的使用日志信息
     ?>
     </td>
    <td align="center" bgcolor="#FFFFFF">
     <?php
     if($myrows['car_log']==null){                   //判断车辆日志是否为空，如果为空，则执行下面的内容
          echo "<a href='indexs.php?lmbs=发货单&ljid=$myrow[id]'>使用该车</a>";
     }else{                                          //如果车辆日志不为空，则执行下面的内容
          echo "<a href='indexs.php?lmbs=发货单&ljid=$myrow[id]'>预定该车</a>";
     }
     ?>
</td>
</tr>
<?php }}else{   echo "您查找的路线不存在!"; }}?>
```

> **说明**
> ❶ select1：指定出发地点。
> ❷ select2：指定到达地点。
> ❸ if 语句：判断从表单中获取的变量是否为真。
> ❹ like：执行模糊查询，从数据库中读取符合条件的数据。

18.6.3 车源信息查询模块的实现过程

车源信息查询模块的功能是根据客户提供的信息，从数据库中读取有关某个路线的车辆使用情形，及时为客户确定一个合理的物流配送方案。该模块的运行结果如图 18.23 所示。



图 18.23 车源信息查询模块的运行结果

车源信息查询功能的实现通过 car_select.php 文件来完成，首先与数据库建立连接；然后创建一个 form 表单，将客户的地址信息提交到当前页，作为查询的条件；最后编写 PHP 脚本语句，以表单中获取的地点数据为条件，执行 like 模糊查询，从数据库的指定表中查询符合条件的信息，并且将查询出的信息显示到页面中。其关键代码可以参考光盘\TM\18\car_select.php 文件。

18.7 发货单管理模块设计

视频讲解：光盘\TM\Video\第 18 章\发货单管理模块设计.exe

18.7.1 发货单管理模块概述

发货单管理模块主要包括发货单的填写、发货单查询、发货单打印和发货单删除。发货单管理模块的主要功能如图 18.24 所示。



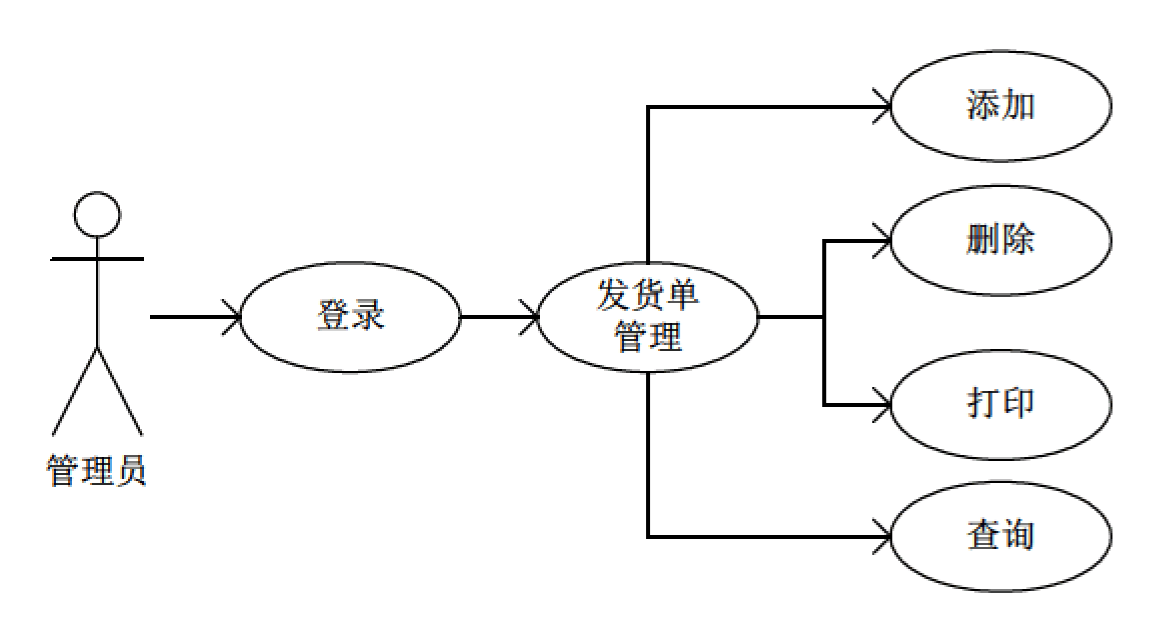


图 18.24　发货单管理模块的功能结构图

18.7.2　发货单编号生成技术

如何才能发挥函数的最大作用？设计者不但能够从表面理解函数的功能和作用，而且还能够以一种发散的思维方式去考虑问题，不要被其本身的功能所束缚，要将函数中某些功能有机地结合起来，从而发挥其更大的作用。

在发货单管理模块中应用到一个发货单的编号，该编号是发货单的唯一标识，不允许存在重复。在编写生成这个编号的程序时，首选方案是应用随机函数获取随机的数字作为编号，这是一个最直接的做法，但是考虑到发货单编号不但具有唯一性，而且还要使其具有一定的标志性，从这个角度考虑使用随机函数就有些不适合，因为其不具备一定的标志性。这时，如果应用 date()函数则可以达到上述所说的要求，不但编号具有唯一性，而且还有规律可循。

为了能够更好地理解这种发散思维的编程思想，首先介绍一下 date()函数。语法格式如下：

```
string date ( string format , int timestamp)
```

该函数返回将参数 timestamp 按照指定格式字符串而产生的字符串，其中的参数 timestamp 是可选的，默认值为 time()，即如果没有给出时间戳则使用本地当前时间。

date()函数的参数 format 的格式化选项如表 18.1 所示。

表 18.1　参数 format 的格式化选项

参　数	说　明
a	小写的上午和下午值，返回值为 am 或 pm
A	大写的上午和下午值，返回值为 AM 或 PM
B	Swatch Internet 标准时，返回值为 000～999
d	月份中的第几天，有前导零的 2 位数字，返回值为 01～31
D	星期中的第几天，文本格式，3 个字母，返回值为 Mon～Sun
F	月份，完整的文本格式，返回值为 January～December
g	小时，12 小时格式，没有前导零，返回值为 1～12
G	小时，24 小时格式，没有前导零，返回值为 0～23
h	小时，12 小时格式，有前导零，返回值为 01～12
H	小时，24 小时格式，有前导零，返回值为 00～23
i	有前导零的分钟数，返回值为 00～59
I	判断是否为夏令时，如果是夏令时返回值为 1，否则为 0
j	月份中的第几天，没有前导零，返回值为 1～31
l	星期数，完整的文本格式，返回值为 Sunday 到 Saturday

续表

参　数	说　明
L	判断是否为闰年，如果是闰年返回值为 1，否则为 0
m	数字表示的月份，有前导零，返回值为 01～12
M	3 个字母缩写表示的月份，返回值为 Jan～Dec
n	数字表示的月份，没有前导零，返回值为 1～12
o	与格林威治时间相差的小时数，如 0200
r	RFC 818 格式的日期，如 Thu，21 Dec 2000 16:01:07 +0200
s	秒数，有前导零，返回值为 00～59
S	每月天数后面的英文后缀，2 个字符，如 st、nd、rd 或者 th，可以和 j 一起使用
t	指定月份所应有的天数
T	本机所在的时区
U	从 UNIX 纪元（January 1 1970 00:00:00 GMT）开始至今的秒数
w	星期中的第几天，数字表示，返回值为 0～6
W	ISO-8601 格式年份中的第几周，每周从星期一开始
Y	4 位数字完整表示的年份，返回值如 1998、2008
y	2 位数字表示的年份，返回值如 88 或 08
z	年份中的第几天，返回值为 0～366
Z	时差偏移量的秒数。UTC 西边的时区偏移量总是负的，UTC 东边的时区偏移量总是正的，返回值为 −43200～43200

注意 这里有效的时间戳典型范围是格林威治时间 1901 年 12 月 13 日 20:45:54 到 2038 年 1 月 19 日 03:14:07（此范围符合 32 位有符号整数的最小值和最大值）。在 Windows 系统中此范围限制为从 1970 年 1 月 1 日到 2038 年 1 月 19 日

技巧 要应用发散的思维方式，在考虑编程方法时就不能被 date()函数自身的功能所限制，date()函数获取的时间不只可以用来记录历史中的某一时刻，而且还可以将其应用到其他的地方，作为一个有规律的随机数。

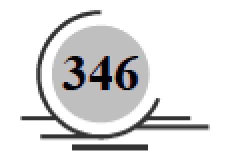
在本项目中的发货单填单模块中就是应用 date()函数获取当前时间和车辆 ID 作为发货单的编号。关键代码如下：

```
<?php
echo date("YmdHis");                                    //获取系统的当前时间
echo $ljid;                                             //获取车辆的 ID
?>"
```

其运行结果如图 18.25 所示。



图 18.25　发货单编号生成的运行结果



技巧　程序开发不只是要看到其表面的功能和属性，更重要的是将某些功能或者属性有机地结合起来，从而开发出更有意义的功能，这才是程序员所要追求的。例如，在学习字符串中的 str_ireplace()函数时，不但要知道该函数的功能是实现字符串的替换功能，而且要学会如何巧妙地运用其实现更好的功能。当换个角度去考虑这个函数的功能时，对这个函数的参数字符串进行颜色改变。这时实现的功能就不只是简单的字符串替换，而且对字符串实现了描红的功能。

18.7.3　发货单填单的实现过程

发货单的填写是客户在确定物流配送路线后填写的一个货物配送详细信息单据，内容包括发货单编号、车牌号码、车主电话、发货人、发货人电话、发货地址、付款方式、收货人、收货人电话、收货地址和配送说明。发货单填单的运行结果如图 18.26 所示。

图 18.26　发货单填单的运行结果

发货单填单的实现通过 insert_dds.php 文件来完成，首先随机生成一个订单编号，然后创建一个 form 表单，最后将表单中的数据提交到数据库中。在 insert_dds.php 文件中使用的重要表单元素如表 18.2 所示。

表 18.2　发货单填单中使用的重要表单元素

名　称	元 素 类 型	重 要 属 性	含　义
form1	form	method="post" action="insert_dd_ok.php" onSubmit="javascript: return check_dd()	表单
fahuo_id	text	id="fahuo_id" value="<?php echo date("YmdHis");?><?php echo $ljid;?>"	发货单编号
car_number	text	value="<?php echo $myrow[car_number]?>"	车牌号码
car_tel	text	value="<?php echo $myrow[tel];?>"	车主电话
fahuo_fk	select	<option selected="selected"></option><option>发货人付款</option><option>第三方付款</option>	付款方式
car_log	textarea	cols="65" rows="5"	说明

将表单中提交的数据存储到数据库是通过 insert_dd_ok.php 文件来完成的，在该文件中首先连接数据库，然后通过 preg_match()函数判断表单中提交的电话号码的格式是否正确，最后将表单中提交的数据存储到 tb_shopping 表中，并且将车辆使用的说明提交到 tb_car_log 表中，将客户信息提交到 tb_customer 表中。其关键代码如下：

（代码位置：光盘\TM\Instance\18\insert_dd_ok.php）

```
<?php
session_start();
if($_SESSION['admin_user']==true){                          //判断登录的用户是否正确
include("conn/conn.php");                                   //连接数据库
$fahuo_time=date("Y-m-d H:i:s");                            //获取系统当前时间
    /* *********************判断表单中提交的电话号码的格式是否正确********************* */
if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$_POST['car_tel'],$countes)){

if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$_POST['fahuo_tel'],$countes)){

if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$_POST['shouhuo_tel'],$countes)){
    /* *********************将表单提交的数据添加到 tb_shopping 表中*********************** */
    $query="insert into tb_shopping (car_number,car_tel,fahuo_id,fahuo_user, fahuo_tel,fahuo_address,
fahuo_content,fahuo_time,fahuo_fk,shouhuo_user,shouhuo_address,shouhuo_tel)values('".$_POST['car_num
ber']."','".$_POST['car_tel']."','".$_POST['fahuo_id']."','".$_POST['fahuo_user']."','".$_POST['fahuo_tel']."','".$_PO
ST['fahuo_address']."','".$_POST['fahuo_content']."','".$_POST['fahuo_time']."','".$_POST['fahuo_fk']."','".$_POS
T['shouhuo_user']."','".$_POST['shouhuo_address']."','".$_POST['shouhuo_tel']."')";
$result=mysql_query($query);
    /* ***********************将车辆使用说明添加到 tb_car_log 表中* ********************** */
  $querys="insert into tb_car_log(car_log,car_number,log_date,fahuo_id)values('".$_POST['car_log']."','".$_POST
['car_number']."','".$_POST['fahuo_time']."','".$_POST['fahuo_id']."')";
  $results=mysql_query($querys);
    /* ***********************将客户信息提交到 tb_customer 表中* ************************ */
  $queryss="insert into tb_customer (customer_user,customer_tel,customer_address)values('".$_POST['fahuo_user']."',
'".$_POST['fahuo_tel']."','".$_POST['fahuo_address']."')";
  $resultss=mysql_query($queryss);
    echo "<script>alert('发货单添加成功!');window.location.href='indexs.php?lmbs=发货单';</script>";
              }else{
                echo "<script>alert('您输入的收货人的电话号码格式不正确!!');history.back()</script>"; }
         }else{
             echo "<script>alert('您输入的发货人的电话号码格式不正确!!');history.back()</script>"; }
    }else{
           echo "<script>alert('您输入的车主的电话号码格式不正确!!');history.back()</script>"; }
}
}else{                                                      //判断用户是否正确登录，如果不是则返回登录页面
    echo "<script>alert('请您正确登录!'); window.location.href='index.php';</script>";}
?>
```

18.7.4　发货单查询的实现过程

发货单查询是为了便于对发货单进行查找以及处理而设计的一个功能，通过其可以准确地查找到指定的发货单，并且还设置一个发货单删除的功能，可以对已经失效或者作废的发货单进行删除。其实现的原理主要是通过以发货单编号为条件的精确查找或者以发货人姓名为条件的模糊查询，然后将查询的结果输出到页面中。运行结果如图 18.27 所示。

发货单查询：	发货单编号	2011122514362913	提交		
发货单编号：	2011122514362913	车牌号码：	吉A-*****	联系电话：	136043*****
发货人：	刘**	电话：	136*****	付款方式：	第三方付款
货物描述：	大米				
发货地址	长春市				
收货人：	李**		收货人电话： 136*****		
收货地址：	沈阳				
发货单处理：	0				
说明：	春送到沈阳，往返3天时间4				
			删除发货单		

图 18.27　发货单查询的运行结果

发货单查询的实现主要通过 hwys.php 文件完成，首先与数据库建立连接，然后创建一个表单



form1，通过该表单来提交查询的条件，可以选择精确查找（以发货单编号为条件）或者模糊查询（以发货人姓名为条件），最后根据 form1 表单中提交的数据，执行查询语句，从数据库中读取出符合条件的发货单的内容。程序的关键代码如下：

（代码位置：光盘\TM\Instance\18\hwys.php）

```
<?php session_start(); include("conn/conn.php");                       //初始化 session 变量，连接数据库
      if($_SESSION['admin_user']==true){                                //判断管理员是否是正确登录
?>
<!-->省略了部分代码<!-->
<?php
      /* *************判断当 form1 表单提交的值为发货单编号时，执行下面的语句* ************* */
      if($_POST['select1']=="发货单编号" and $_POST['select1']!=""){
      $query="select * from tb_shopping where fahuo_id='".$_POST['select1']."'";
      $result=mysql_query($query);                                      //执行查询语句
      $myrow=mysql_fetch_array($result);                                //获取查结果
      }
      /* ****************如果 form1 表单提交的值为发货人时，执行下面的语句* **************** */
      if($_POST['select1']=="发货人" and $_POST['select1']!=""){
      $query="select * from tb_shopping where fahuo_user='".$_POST['select1']."'";
      $result=mysql_query($query);                                      //检索指定发货人的信息
      $myrow=mysql_fetch_array($result);
      }
      ?>
… //省略部分 HTML 代码
<td colspan="4" bgcolor="#FFFFFF"><textarea name="car_log">
<!-- ---------------------------------------输出车辆日志中的车辆使用情况------------------------------------------- -->
<?php
      $querys="select * from tb_car_log where fahuo_id='$myrow[fahuo_id]'";     //从车辆日志表中查找指定的数据
      $results=mysql_query($querys);                                        //执行查询语句
      $myrows=mysql_fetch_array($results);                                  //获取查询结果
      echo $myrows['car_log'];                                              //输出车辆日志信息
      ?>
      </textarea></td>
      <!-- ---------------------------根据发货单的 ID 设置一个发货单删除的超链接---------------------------- -->
      <td bgcolor="#FFFFFF"><a href="fhd_qr.php?delete=<?php echo $myrow['id'];?>">删除发货单</a></td>
… //省略部分 HTML 代码
<?php }else{          //判断管理员是否正确登录，如果错误则返回到登录页
      echo "<script>alert('请您正确登录!'); window.location.href='index.php';</script>";
}
?>
```

发货单删除的操作通过 fhd_qr.php 文件来完成，主要根据超链接中提供的发货单 ID 执行删除的操作。其关键代码如下：

（代码位置：光盘\TM\Instance\18\fhd_qr.php）

```
<?php session_start(); include("conn/conn.php");              //初始化 session 变量，连接数据库
if($_SESSION['admin_user']==true){
      if($_GET['fahuo_id']==true){                            //判断提交的发货单 ID 是否为真
            $query="delete from tb_car_log where fahuo_id='".$_GET['fahuo_id']."'";
                                                              //根据发货单 ID 执行删除发货单的操作
            $result=mysql_query($query);
            $query="update tb_shopping set fahuo_ys='1' where fahuo_id='".$_GET['fahuo_id']."' ";
            $result=mysql_query($query);
            echo  "<script>alert('发货回执单处理成功!');window.location.href='indexs.php?lmbs=回执发货单确认
';</script>";
      }
      if($_GET['delete']==true){
            $query="delete from tb_shopping where id='".$_GET['delete']."'";
```

```
            $result=mysql_query($query);
            /* **********************发货单删除成功，弹出提示，并返回到上一页* ********************* */
            echo "<script>alert('发货单删除成功!');window.location.href='indexs.php?lmbs=发货单查询';</script>";
        }
}else{
        /* **********************发货单删除失败，弹出提示，并返回到上一页* ************************ */
        echo "<script>alert('请您正确登录!'); window.location.href='index.php';</script>";
}
?>
```

18.7.5　发货单打印的实现过程

发货单的打印是为货物的发送提供一个书面的依据，作为企业与客户商业活动的凭证，其内容必须真实、准确。发货单打印的运行结果如图 18.28 所示。

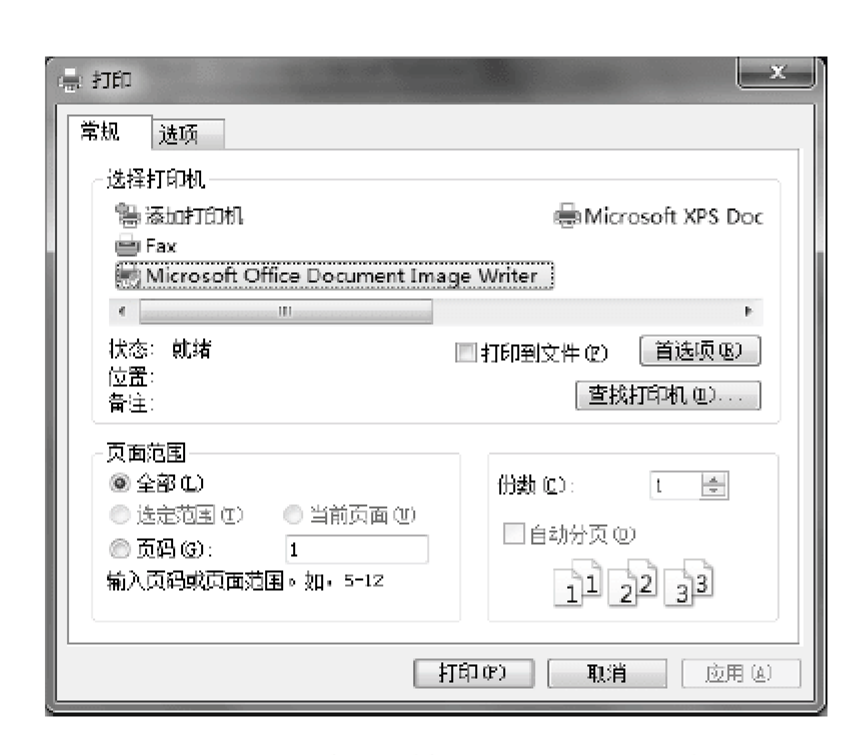


图 18.28　发货单打印的运行结果

这里的发货单打印技术主要通过框架打印技术来完成，即只打印框架中的内容，而网页中其他的内容不打印。实现原理是：首先在 insert_dd.php 文件中创建一个浮动框架，设置要指定打印内容的范围，并且连接到要打印的 insert_dds.php 文件，然后在指定的文件中输出要打印的内容，最后应用 onclick 事件调用 parent.content.focus()和 window.print()方法实现打印功能。

在 insert_dd.php 文件中创建一个浮动框架，设置框架名称为 content，设置打印的范围宽 97%，高 252%，连接到要打印的 insert_dds.php 文件。程序代码如下：

（代码位置：光盘\TM\Instance\18\insert_dd.php）

```
<iframe name="content" src="insert_dds.php?ljid=<?php echo $ljid;?>"   frameborder="0" width="97%"
height="252%"></iframe>
```

设置要打印的内容，通过 insert_dds.php 文件来完成，发货单填写的内容以表单的形式显示，其中的内容可以参考光盘中 TM\18\insert_dds.php 文件，这里不再赘述。

最后回到 insert_dd.php 页中，在该页中设置一个超链接，应用 onClick 事件调用 parent.content.focus()和 window.print()方法实现发货单的打印功能。程序关键代码如下：

（代码位置：光盘\TM\Instance\18\insert_dd.php）

```
<a href="#"
      onClick="parent.content.focus();window.print();">
            <img src="images/dl_033.gif" width="87" height="27" border="0">
</a>
```

18.8　回执单验收管理模块设计

视频讲解：光盘\TM\Video\第 18 章\回执单验收管理模块设计.exe

18.8.1　回执单模块概述

回执单模块的主要功能就是对货物配送完成确认。将该发货单执行类型更新为“1”，表明该次物流配



送已经完成；清空该车辆的使用日志，便于执行下一个订单。该模块的业务操作流程如图 18.29 所示。

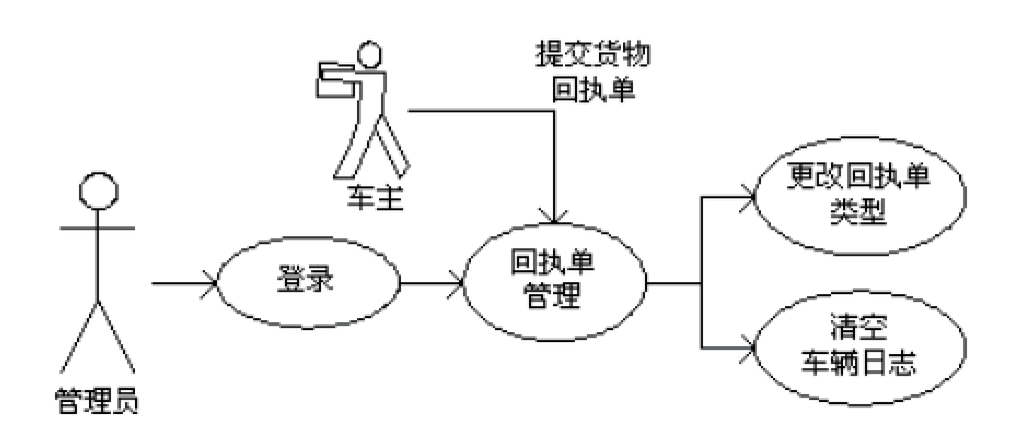


图 18.29　回执单模块的业务操作流程

18.8.2　MySQL 函数库函数应用技术

在回执单模块中，要对回执单进行确认首先要查找到指定的发货单，并且对其内容进行核实。这里应用的是根据发货单的编号进行查询，从数据库中读取出符合条件的数据。在执行查询语句、从数据库中读取数据时，主要应用 mysql_query()、mysql_num_rows()和 mysql_fetch_array()函数。

首先通过 mysql_query()函数执行一个 select 查询语句，然后通过 mysql_num_rows()函数判断查询的结果是否有值，如果没有值则输出“您查找的发货单编号不存在！”，如果有值则执行 mysql_fetch_arrau()函数，输出数据库中查询到的数据。

mysql_query()函数向与指定的连接标识符关联的服务器的当前活动数据库中发送一条 MySQL 查询。语法如下：

```
resource mysql_query ( string query [, resource link_identifier] )
```

mysql_query()函数的参数 query 是字符串的类型，传入的是 SQL 指令；参数 link_identfier 是资源类型，传入的是由 mysql_connect()函数或者 mysql1_pconnect()函数返回的连接号。如果省略该参数，则会使用最后一个打开的 MySQL 数据库连接。



查询字符串不应以分号结束。

mysql_num_rows()函数获取结果集中行的数目。语法如下：

```
int mysql_num_rows ( resource result )
```



该函数仅对 SELECT 语句有效。

mysql_fetch_array()函数返回根据从结果集获取的行生成的数组，如果没有更多行则返回 false。语法如下：

```
array mysql_fetch_array ( resource result [, int result_type] )
```

mysql_fetch_array()函数的参数说明如表 18.3 所示。

表 18.3　mysql_fetch_array()函数的参数说明

参 数 名 称	说　　明
result	资源类型的参数，要传入的是由 mysql_query()函数返回的数据指针
result_type	可选参数，整数型参数，要传入的是 MYSQL_ASSOC、MYSQL_NUM 和 MYSQL_BOTH 3 种由 PHP 定义好的常数之一，默认值是 MYSQL_BOTH 用 MYSQL_ASSOC 只得到关联索引（相当于 mysql_fetch_assoc()函数） 用 MYSQL_NUM 只得到数字索引（相当于 mysql_fetch_row()函数） 用 MYSQL_BOTH 将得到一个同时包含关联和数字索引的数组



本函数返回的字段名是区分大小写的。

通过上述 3 个函数就可以完成从数据库中读取数据，并且将数据输出到页面的功能。

注意 使用该种方法输出的只是数据库中的一条数据。如果加上 while 循环语句，那么就可以完成数据库中所有符合条件数据的循环输出。

应用 while 循环语句读取数据库中的数据，这里以读取 tb_shopping 表中的数据为例。代码如下：

```
<?php include("conn/conn.php");                                    //连接数据库
     $query="select * from tb_shopping ";                          //编写查询语句
     $result=mysql_query($query);                                  //执行查询语句
❶    if(mysql_num_rows($result)<1){                               //判断数据库中是否存在数据
     echo "您查找的内容不存在!";
     }else{
❷    while($myrow=mysql_fetch_array($result)){                    //循环输出数据库中的数据
?>
<?php
 echo $myrow[id];                                                  //输出数据库中数据
?>
…                                                                  //输出内容的代码部分略
<?php }?>
```

说明 ❶ mysql_num_rows()：返回根据从结果集获取的行生成的数组，如果没有更多行则返回 false。❷ mysql_fetch_array()：获取结果集中行的数目。该函数只对 select 语句有效。

18.8.3 回执单验收模块的实现过程

回执单验收模块的主要功能就是对货物配送完成确认，运行结果如图 18.30 所示。



图 18.30 回执单验收模块的运行结果

回执单验收模块的实现过程是：首先在 select_dhd.php 页中根据发货单的编号查询出指定的发货单，对发货单进行核实，然后单击“发货订单确认”超链接，实现回执单的验收。其关键代码如下：

（代码位置：光盘\TM\Instance\18\select_dhd.php）

```
<?php
     include("conn/conn.php");                                           //连接数据库
          if($_POST['Submit']==true){
          $query="select * from tb_shopping where fahuo_id='".$_POST['select']."'";     //编写查询语句
          $result=mysql_query($query);                                   //执行查询语句
          if(mysql_num_rows($result)<1){                                 //判断数据库中是否存在数据
               echo "您查找的发货单编号不存在！";
          }else{
               $myrow=mysql_fetch_array($result);                        //输出数据库中的数据
     ?>
…                                                                        //输出内容的代码部分略
     <td colspan="4" bgcolor="#FFFFFF">
          <textarea name="car_log">
               <?php
                    /* **************根据提交的货物编号，查询车辆日志中的对应日志************** */
                    $querys="select * from tb_car_log where fahuo_id='$myrow[fahuo_id]'";
                    $results=mysql_query($querys);               //执行查询语句
                    $myrows=mysql_fetch_array($results);         //输出数据库中的数据
```

```
				echo $myrows[car_log];				//输出数据库中的数据
			?>
		</textarea></td>
	...									//输出内容的代码部分略
	<!-- ----------------------------------------设置超链接，执行发货单的确认------------------------------------- -->
	<td bgcolor="#FFFFFF"><a href="fhd_qr.php?fahuo_id=<?php echo $myrow[fahuo_id];?>">发货订单回执
确认</a></td>
```

最后在 fhd_qr.php 页根据超链接提供的发货单编号更新发货单类型为“1”，清空对应车辆的使用日志。其程序的关键代码如下：

（代码位置：光盘\TM\Instance\18\fhd_qr.php）

```
<?php session_start(); include("conn/conn.php");				//初始化 session 变量
if($_SESSION['admin_user']==true){						//判断用户是否是正确登录
	if($_GET['fahuo_id']==true){					//判断超链接传递的变量值是否为真
		/* *****************以超链接传递的变量值为条件，编写删除语句******************* */
		$query="delete from tb_car_log where fahuo_id='".$_GET['fahuo_id']."'";
		$result=mysql_query($query);				//执行删除语句
		/* ********************更新发货单数据表中发货的类型为“1” *********************** */
		$query="update tb_shopping set fahuo_ys='1' where fahuo_id='".$_GET['fahuo_id']."'"
;
		$result=mysql_query($query);
			echo "<script>alert('发货回执单处理成功!');window.location.href='indexs.php?lmbs=回执发货单
确认';</script>"; }
	}else{
		echo "<script>alert('请您正确登录!'); window.location.href='index.php';</script>";
	}
?>
```

18.9　基础信息管理模块设计

视频讲解：光盘\TM\Video\第 18 章\基础信息管理模块设计.exe

18.9.1　基础信息管理模块概述

基础信息管理模块主要包括客户信息管理，用于添加客户信息和删除客户信息；车源信息管理，主要用于对车源信息进行管理，实现车源信息的添加、修改和删除功能；管理员信息管理，用于修改管理员的密码。基础信息管理模块的功能结构如图 18.31 所示。

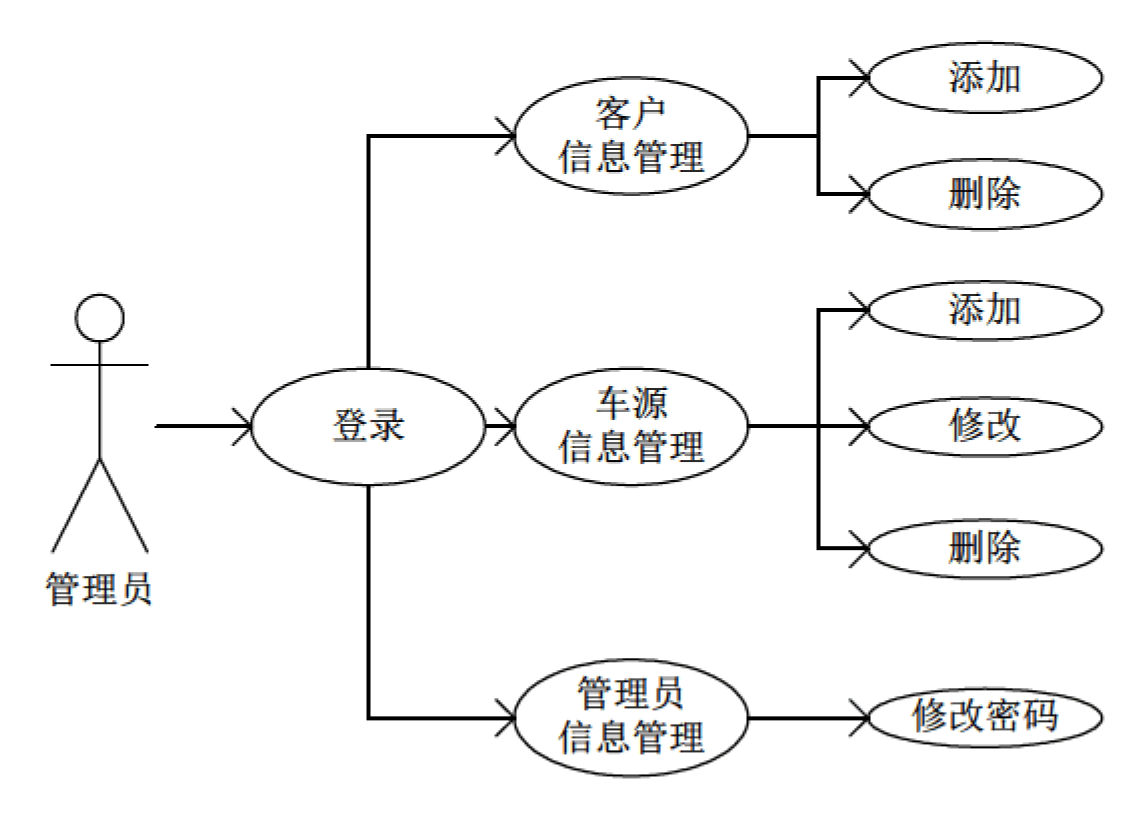


图 18.31　基础信息管理模块的功能结构图

18.9.2　面向对象封装密码修改类

修改管理员密码的模块中，在对密码进行处理的过程中用的是面向对象技术实现密码的修改。

面向对象技术的关键就是类的创建，首先创建一个 chkuser 类，其中定义 5 个数据成员，实现对管理员密码的修改，分别是管理员名称（$admin_user）、

管理员密码（$admin_pass）、管理员新密码（$admin_new_pass）、管理员新密码确认（$admin_new_pass2）和验证码（$yzm），然后应用 chkuser 函数实现数据成员的初始化，最后应用数据成员函数 chkinput 实现密码的修改。程序代码如下：

（代码位置：光盘\TM\Instance\18\update_pass_ok.php）

```
<?php   session_start();                                    //初始化 session 变量
❶    class chkuser{                                         //创建一个 chkuser 类
            var $admin_user;                                //创建数据成员，用户名
            var $admin_pass;                                //创建数据成员，密码
            var $admin_new_pass;                            //创建数据成员，新密码
            var $admin_new_pass2;                           //创建数据成员，密码确认
            var $yzm;                                       //创建数据成员，验证码
      function chkuser($x,$y,$m,$n,$z){                     //创建 chkuser 函数，实现数据成员初始化
      $this->admin_user=$x;
      $this->admin_pass=$y;
            $this->admin_new_pass=$m;
            $this->admin_new_pass2=$n;
            $this->yzm=$z;
      }
❷    function chkinput(){                                   //创建 chkinput 函数实现密码更改
            //判断输入的验证码是否正确
❸          if(strval($this->yzm)!=$_SESSION["autonum1"]){
            echo "<script>alert('验证码输入错误!');history.back();</script>";
            exit;
            }
      include("conn/conn.php");                             //连接数据库
                                                            //判断输入的密码是否正确
            $query=mysql_query("select * from tb_admin where admin_user='".$this->admin_user."' and admin_
pass='".$this->admin_pass."'");
            if(mysql_num_rows($query)<1){
                  /* *********************如果密码不正确则提示重新输入密码********************* */
                  echo "<script>alert('您输入的密码不正确，请重新输入');history.back();</script>";
            }else{
                  /* *****************如果密码正确，则应用更新语句，实现密码的更改*************** */
                  $query1=mysql_query("update tb_admin set admin_pass='".$this->admin_new_pass."'");
                  if($query1==true){
                        echo "<script>alert('密码更改成功');history.back();</script>";
                  }
            }
      }
      }
      /* ***********************************类的实例化过程************************************* */
❹    $chk=new
chkuser($_POST[admin_user],md5($_POST[admin_pass]),md5($_POST[admin_new_pass]),$_POST[admin_
new_pass2],$_POST[yzm]);
      $chk->chkinput();
?>
```

说明

❶ chkuser：创建一个类 chkuser，用于修改管理员密码。

❷ chkinput()：创建一个自定义函数 chkinput()，用于实现密码修改。

❸ strval()：获取变量的字符串。

❹ new chkuser：实现类的实例化，应用类实现密码修改。

18.9.3　客户信息管理的实现过程

客户信息管理模块的主要功能就是向数据库中添加客户信息，并且可以对指定的客户进行删除操作。客户信息管理模块的的运行结果如图 18.32 所示。

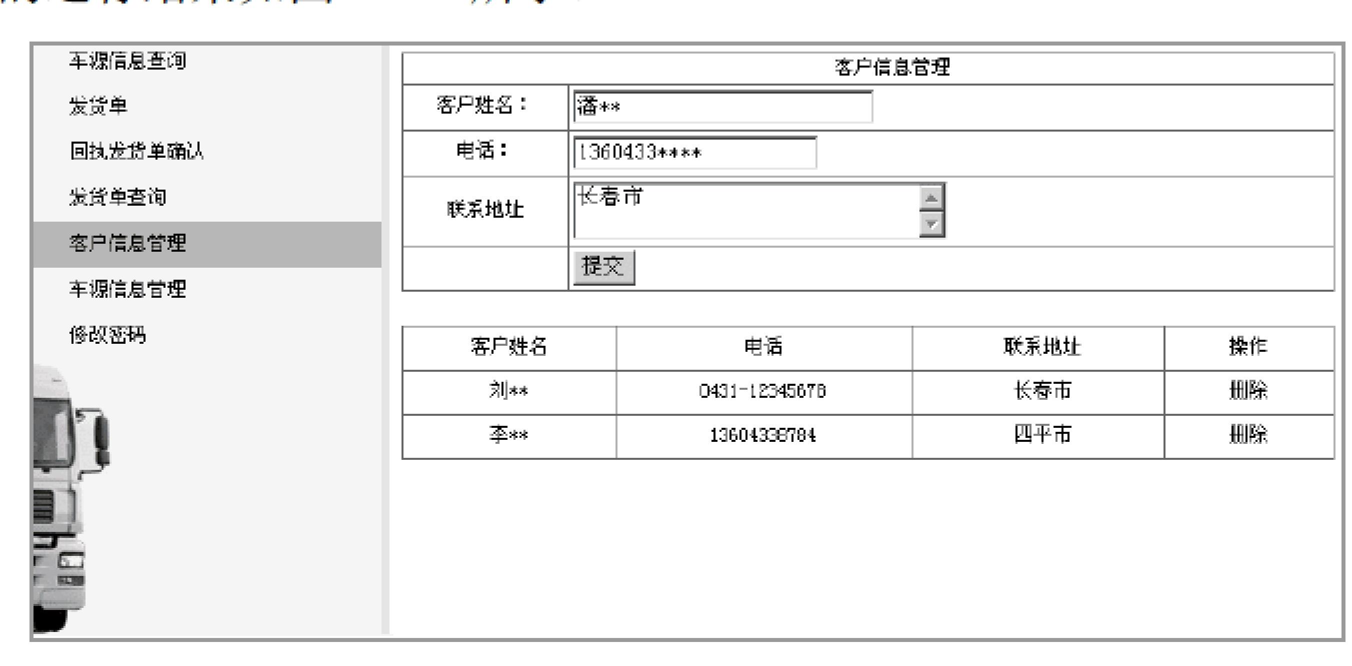


图 18.32　客户信息管理模块的运行结果

客户信息管理模块主要通过 customer.php、customer_ok.php 和 delete_customer.php 文件来完成。首先通过 customer.php 文件来添加客户信息，并且输出数据库中的所有客户信息，设置删除客户信息的超链接。其输出客户详细信息的关键代码如下：

（代码位置：光盘\TM\Instance\18\customer.php）

```
<?php include("conn/conn.php");                              //连接数据库
      $query=mysql_query("select * from tb_customer ");       //执行查询语句，从数据库中读取数据
      while($myrow=mysql_fetch_array($query)){               //通过 while 循环语句，循环输出数据库中的数据
?>
<tr>
     <td height="27" align="center" bgcolor="#FFFFFF"><?php echo $myrow['customer_user'];?></td>
     <td align="center" bgcolor="#FFFFFF"><?php echo $myrow['customer_tel'];?></td>
     <td align="center" bgcolor="#FFFFFF"><?php echo $myrow['customer_address'];?></td>
      <!-- ---------------设置删除客户信息的超链接，并且设置客户的 ID 为变量--------------------- -->
     <td align="center" bgcolor="#FFFFFF"><a href="delete_customer.php?delete=<?php echo
$myrow['customer_id'];?>">删除</a></td>
</tr>
   <?php }?>
```

客户信息添加的操作通过 customer_ok.php 文件完成，首先连接数据库，然后判断提交的电话号码格式是否正确，最后将客户信息数据提交到数据库中。程序的代码如下：

（代码位置：光盘\TM\Instance\18\customer_ok.php）

```
<?php include("conn/conn.php");                                                        //连接数据库
      if($_POST['Submit']==true and $_POST['customer_user']==true) {          //判断提交的值是否为真
            /* ***************************判断提交的电话号码格式是否正确*************************** */

      if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$_POST['customer_tel'],$cou
ntes)){
                  /* **************************将提交的数据添加到客户信息表中************************ */
                  $query=mysql_query("insert  into  tb_customer(customer_user,customer_tel,customer_address)
values('".$_POST['customer_user']."','".$_POST['customer_tel']."','".$_POST['customer_address']."')");
                  if($query==true){                                     //如果添加操作成功，弹出提示，并返回首页
                        echo "<script> alert('客户信息添加成功');window.location.href='indexs.php?lmbs=客户信息
管理';</script>";
```

```
            }
        }else{                                          //如果添加操作失败，弹出提示，并返回上一页
        echo "<script>alert('您输入的电话号码格式不正确!!');history.back()</script>";
    }
}
?>
```

客户信息删除的操作通过 delete_customer.php 文件来完成，主要以超链接中提交的变量为根据，删除数据库中指定的客户信息。程序的关键代码如下：

（代码位置：光盘\TM\Instance\18\delete_customer.php）

```
<?php include("conn/conn.php");                         //连接数据库
    if($_GET['delete']==true){                          //判断提交的变量值是否为真
        /* ********************根据$_GET 获取的变量值为依据，执行删除语句******************** */
        $query=mysql_query("delete from tb_customer  where customer_id='".$_GET['delete']."'");
        if($query){                                     //如果删除操作成功，则弹出提示，并返回首页
            echo  "<script>  alert('客户信息添加成功');window.location.href='indexs.php?lmbs=客户信息管理
';</script>";
        }
    }
?>
```

18.9.4　车源信息管理的实现过程

车源信息管理模块主要实现对车辆信息的添加、修改和删除操作。车源信息管理模块的运行结果如图 18.33 所示。

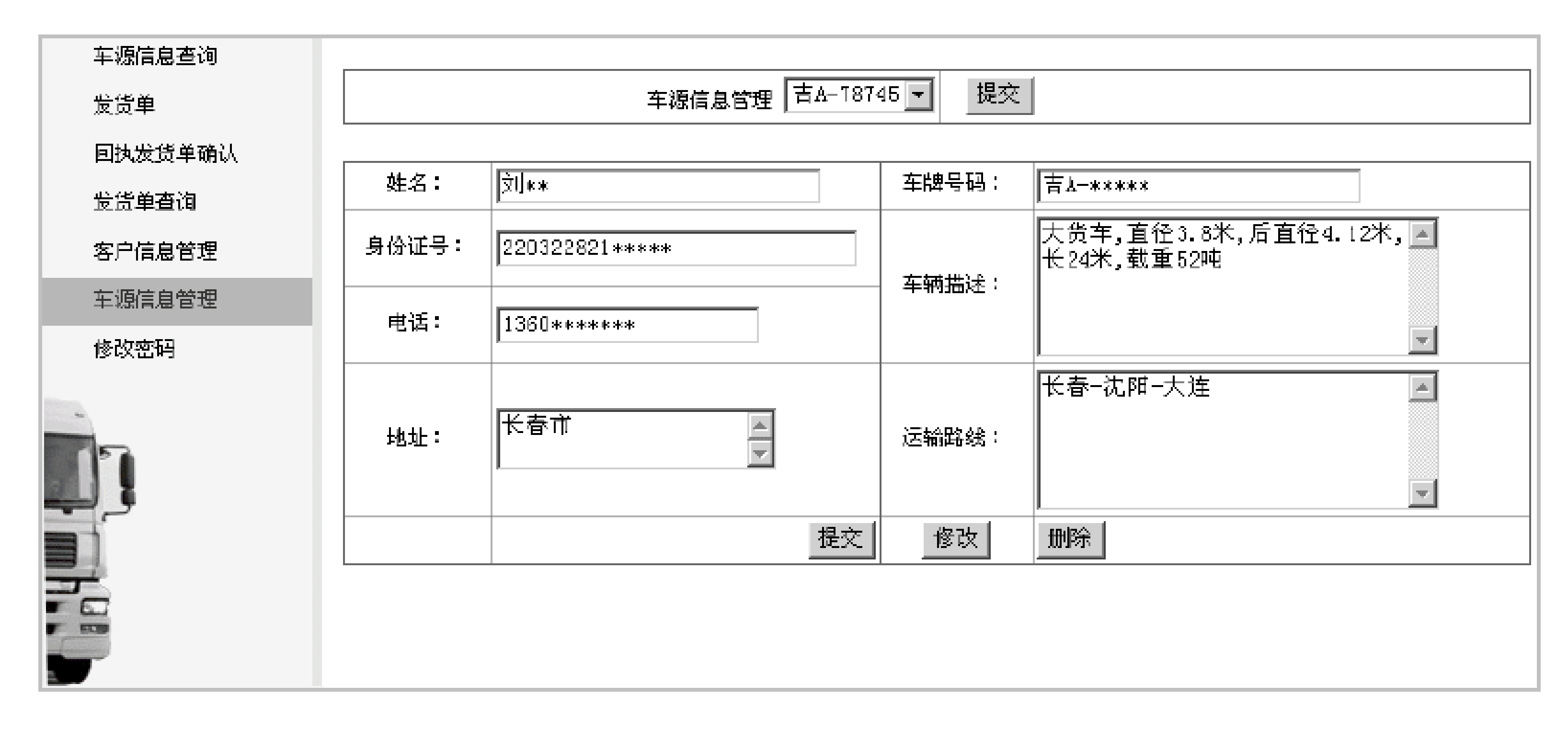
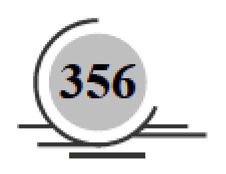


图 18.33　车源信息管理模块的运行结果

车源信息管理模块主要由两个文件组成。一个是 car.php 文件，以表单文件为主用于输出车辆的信息和提交车辆的信息。设置 form2 表单，根据车牌号码从数据库中读取对应车辆的信息，进而实现对该车辆信息的修改或者删除操作；设置 form1 表单，用于显示从数据库中读取的车辆信息和直接添加车辆信息到数据库。form1 表单中使用的重要元素如表 18.4 所示。

表 18.4　车源信息管理模块中使用的重要表单元素

名　称	元 素 类 型	重 要 属 性	含　义
form1	form	method="post" action="car_insert_ok.php" onSubmit="javascript:return check_car(form1);"	表单
username	text	id="username" value="<?php echo $myrows[username];?>"	车主姓名



续表

名 称	元素类型	重要属性	含 义
car_number	text	id="car_number" value="<?php echo $myrows[car_number];?>"	车牌号码
car_tel	text	value="<?php echo $myrow[tel];?>"	车主电话
user_number	text	id="user_number" value="<?php echo $myrows[user_number];?>"	身份证号码
car_content	textarea	<?php echo $myrows[car_content];?>	车辆描述
address	textarea	<?php echo $myrows[address];?>	家庭地址
car_road	textarea	<?php echo $myrows[car_road];?>	运输路线
Submit	submit	value="提交"	提交表单
Submit2	submit	value="修改"	提交表单
Submit4	submit	value="删除"	提交表单

另一个是 car_insert_ok.php 文件，通过该文件实现对表单中提交的数据进行处理。首先连接数据库，然后根据表单中提交的值进行判断，当按钮$Submit 的值为“提交”时，将提交的数据存储到指定的数据表中；当按钮$Submit2 的值为“修改”时，则更新语句，对车辆的信息进行更新；当按钮$Submit4 的值为“删除”时，则执行删除语句，将指定的车辆信息删除。程序关键代码如下：

（代码位置：光盘\TM\Instance\18\car_insert_ok.php）

```
<?php session_start(); include("conn/conn.php");                //初始化 session 变量，连接数据库
❶    if($_SESSION[admin_user]==true){                         //判断管理员是否是正常登录
          if($_POST['Submit'==true){                          //判断表单提交的值是否为"提交"
               /* ********************判断表单中提交的电话号码的格式是否正确******************** */
❷         if(preg_match("/\d{17}[\d|X]|\d{15}/",$_POST['user_number'],$counts)){

if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$_POST['user_tel'],$countes)){
               /* ************************将表单中的数据添加到 tb_car 数据表中******************** */
               $query=mysql_query("insert into tb_car(username,user_number,tel,address,car_number,car_road,
car_content)values('".$_POST['username']."','".$_POST['user_number']."','".$_POST['user_tel']."','".$_POST['ad
dress']."','".$_POST['car_number']."','".$_POST['car_road']."','".$_POST['car_content']."')");
                    if($query==true){
                         echo  "<script>alert('车源信息添加成功!');window.location.href='indexs.php?lmbs=车
源信息管理';</script>"; }
               }else{
                    echo "<script>alert('您输入的电话号码格式不正确!!');history.back()</script>"; }
          }else{
                    echo "<script>alert('您输入的身份证号码的格式不正确!!');history.back()</script>";     }
          }
          /* ******************************判断表单提交的值是否为"修改"************************** */
❸         if($_POST['Submit2']=="修改"){
               /* **************************判断电话号码的格式是否正确************************** */
          if(preg_match("/\d{17}[\d|X]|\d{15}/",$_POST['user_number'],$counts)){

     if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$_POST['user_tel'],$countes))
{
               /* ***************************更新 tb_car 数据表中的数据*************************** */
                    $query="update tb_car set username='".$_POST['username']."', user_number= '".$_POST['user_
number']."', tel='".$_POST['user_tel']."', address='".$_POST['address']."', car_number= '".$_POST['car_number']."',
car_road='".$_POST['car_road']."',   car_content='".$_POST['car_content']."'   where   car_number='".$_POST
['car_number']."'";
                    $result=mysql_query($query);
                    if($result==true){
```

```
                              echo  "<script>alert('车源数据更新成功!!');window.location.href='indexs.php?lmbs=车
源信息管理';</script>";
                        }
                    }else{
                        echo "<script>alert('您输入的电话号码格式不正确!!');history.back()</script>";
                }
            }else{
                echo "<script>alert('您输入的身份证号码的格式不正确!!');history.back()</script>";
            }
        }
            /* ***************************判断表单提交的值是否为"删除"*************************** */
❹           if($_POST['Submit4']=="删除"){
                $query="delete  from tb_car where car_number='".$_POST['car_number']."'";
            $result=mysql_query($query);
            if($result==true){
                echo  "<script>alert('车源数据删除成功!');window.location.href='indexs.php?lmbs=车源信息管理
';</script>";
            }
        }
}else{
        echo "<script>alert('请您正确登录！ '); window.location.href='index.php';</script>";
}
?>
```

说明

❶ $_SESSION[admin_user]：判断用户是否是正常登录。

❷ preg_match()：应用 preg_match()验证电话号码的格式是否正确。preg_match()对正则表达式进行匹配。返回匹配的次数，0 次（没有匹配）或者 1 次，因为 preg_match()函数在第一次匹配后将停止搜索，出错返回 false。

❸ $Submit2=="修改"：判断表单提交的值是否等于修改，如果等于修改则执行修改的操作。

❹ $Submit4=="删除"：判断表单提交的值是否等于删除，如果等于删除则执行删除语句。

18.10 小　　结

本章从某物流配送公司的实际需求角度出发，开发一个完整的物流配送信息管理系统，详细地讲解了物流管理系统的开发流程，从最初的需求分析、可行性分析，到系统的设计、数据库的设计，其中重点突出车源信息查询模块、发货单管理模块和回执单验收管理模块的设计。通过本章的学习，读者不但可以了解物流配送系统开发的整体思路，而且还能够掌握很多关键的技术和技巧：函数的灵活运用、MySQL 存储过程的创建和应用、报表打印技术的实现、应用正则表达式验证电话号码等。



进阶提高

第19章

ADODB 类库

（视频讲解：64 分钟）

ADODB 提供了一套完整的方法和属性让开发者控制数据库系统。在操作过程中，无须计较使用的是什么数据库，而只要记得它的功能即可。因为在更换数据库时，只要修改一个属性值，ADODB 就会自动依据设定取用正确的 PHP 函数。此外，有可能还需要修改 SQL 语句，因为不同的数据库在 SQL 语句的使用上有一点区别。这就是 ADODB 的魅力所在。

通过阅读本章内容，你可以：

- 了解 ADODB 类库是什么
- 了解 ADODB 支持的数据库
- 熟悉 ADODB 类库的安装
- 熟悉 ADODB 类库的常用方法
- 熟悉 ADODB 类库的应用
- 掌握如何灵活运行 ADODB 类库

19.1　ADODB 类库是什么

ADODB（Active Data Objects Data Base）是存取数据库所使用的一组函数。虽然 PHP 是构建 Web 系统强有力的工具，但是由于 PHP 存取函数一直没有标准化，不同数据库间的函数名称、参数差异很大，在更换数据库时，会带来大量的代码修复工作。这时，就需要一组函数库来隐藏不同数据库函数间的差异，让开发者可以很简单地切换数据库，这就是 ADODB 类库。

ADODB 类库有以下优点：

☑　安装简单，易学易用。提供了与微软的 ADODB 相似的语法功能。无论是初学者，还是从其他语言转过来的开发人员（主要是 ASP），都可以很快上手。
☑　以标准的 SQL 语句书写的程序在更换数据库时，不需要改变源程序。
☑　可以生成 Smarty 循环需要的二维数组，简化了 Smarty 开发。
☑　支持缓存查询，最大可能地提高查询速度。

ADODB 类库的缺点是执行效率慢。由于 ADODB 类库非常庞大，仅是主执行类（adodb.inc.php）就有 125KB，所以在使用 ADODB 类库时要充分考虑到这一点。

19.2　ADODB 支持的数据库

ADODB 目前支持 MySQL、Oracle、Microsoft SQL Server、Sybase、PostgreSQL、FoxPro、Access、ADO 和 ODBC 等大多数数据库。ADODB 所支持的常见数据库及参数如表 19.1 所示。

表 19.1　ADODB 支持的数据库及参数

数据库名称	测试状态	资料库	RecordCount()支持	需要安装的驱动	操作系统
access	B	Microsoft Access/Jet，需要建立一个 ODBC/DSN	Y/N	ODBC	Windows
db2	C	DB2，可以通过 ODBC 获得可以信赖的运作效果	Y/N	DB2 CLI/ODBC interface	UNIX Windows
vfp	A	Microsoft Visual FoxPro，需要建立一个 ODBC/DSN	Y/N	ODBC	Windows
mssql	A	在日期格式上仍有一些问题	Y/N	Mssql client	UNIX Windows
mysql	A	MySQL 支持交易处理	Y/N	MySQL client	UNIX Windows
mysqlt 或 maxsql	A	MySQL 支持交易处理	Y/N	MySQL client	UNIX Windows
oci8	A	Oracle 8/9，支持比 Oracle 驱动程式还多的功能。在连接之前，可能需要先配好环境变数('ORACLE_HOME=...')	Y/N	Oracle client	UNIX Windows

续表

数据库名称	测试状态	资料库	RecordCount()支持	需要安装的驱动	操作系统
oci8po	A	Oracle 8/9 可携式驱动程式	Y/N	Oracle client	UNIX Windows
odbc	A	标准 ODBC 用 PConnect('DSN','user','pwd').连接	? depends on database	ODBC	UNIX Windows
odbc_mssql	C	用 ODBC 连接 MSSQL	Y/N	ODBC	UNIX Windows
odbc_oracle	C	用 ODBC 连接 ORACLE	Y/N	ODBC	UNIX Windows
postgres	C	PostgreSQL 不支援 LIMIT 指令	Y	PostgreSQL client	UNIX Windows
sybase	C	Sybase	Y/N	Sybase client	UNXI Windows

表 19.1 说明如下：

☑ A 表示已经经过很多验证及测试，可靠度最高。

☑ B 表示已经测试并使用，但可能仍有一些功能没有达成。

☑ C 表示使用者自行配置或试用的驱动程式，可能没有完全支持 ADODB 的功能。

☑ “RecordCount()支持”是指 RecordCount()函数是否会回传用 SELECT 指令取得的记录数（不支持传回-1）。Y/N 表示当全局变量$ADODB_COUNTER=true 时，会以模拟的方式取得，但如果估计到记录数会很大时，最好把这个值设为 false，也就是关掉模拟功能，否则会耗掉很多内存。该变量在每次执行时都会检查，所以可以选择性地使用。

19.3 ADODB 下载与安装

要使用 ADODB 来操作数据库，首先就要获取和安装 ADODB。读者只要到网上下载 ADODB 包，解压到 Web 服务器目录下即可。本书使用的 ADODB 的版本是 5.0。



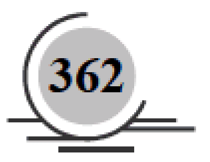
要使用 ADODB，使用的 PHP 必须是 4.0.1 以上的版本。

在详细介绍 ADODB 类库之前，首先了解一下使用 ADODB 类库的简要步骤。

例 19.1 测试 ADODB 是否配置成功。应用 ADODB 类库连接 MySQL 数据库，指定连接的数据库为 db_database19，使用的数据表是 tb_object。将 ADODB 类库存储在根目录的上级目录中，名称为 adodb5。**（实例位置：光盘\TM\Instance\19\19.1）**

代码如下：

```
<?php
include_once ('../adodb5/adodb.inc.php');                              //载入 adodb.ini.php 文件
$conn = ADONewConnection('mysql');                                     //连接 MySQL 数据库
$conn -> PConnect('localhost','root','111','db_database19');           //连接 db_database19 库
$conn -> execute('set names gb2312');                                  //设置编码
$rst = $conn -> Execute("select * from tb_object") or die('执行错误');  //执行 SQL 语句
while(!$rst -> EOF){                                                   //配合 while 语句循环输出结果
```



```
        echo $rst -> fields[1].'<br>';
        $rst -> movenext();                                    //指针下移
}
$rst -> close();
$conn -> close();
?>
```

结果为：图书开发 网络 电器。

使用 ADODB 类库的一般操作步骤为：

（1）载入 adodb.inc.php 文件。要使用 ADODB 类库，首先要启动 ADODB，启动 ADODB 的方法就是载入 adodb.ini.php 文件。

（2）建立连接。使用 ADONewConnection()函数。

（3）连接数据库。使用 PConnect()函数。

（4）执行数据库操作。使用 Execute()方法返回结果集。

（5）对结果集进行操作。

（6）关闭连接。

19.4　ADODB 类库

ADODB 类库非常庞大，仅一个 adodb.inc.php 文件就有 120KB，所以这里只介绍一些常用函数和变量。这些函数按照功能的不同可分为连接数据库函数、操作数据库函数、结果集处理函数和错误处理函数等。

19.4.1　连接数据库函数

视频讲解：光盘\TM\Video\第 19 章\连接数据库函数.exe

在介绍连接数据库的方法之前，先来了解一些在连接数据库过程中应用到的函数。

☑ ADONewConnection($databaseType)

函数作用：连接数据库系统。

参数$databaseType 表示要连接的数据库系统的名称，如 mysql、mssql 等。

☑ PConnect($host,[$user],[$password],[$database])

函数作用：持久化连接数据库。

参数说明如下。

- $host：数据库系统的服务器所在地址。如果是本机操作，格式为 localhost。
- $user：数据库用户名。
- $password：数据库密码。
- $database：使用到的数据库。

Connect()也是连接数据库的函数。该函数为非持久化连接数据库。持久化连接和非持久化连接的区别在于：持久化连接不用每次都创建新连接，可以提高程序的执行速度。但有些数据库不支持，如果遇到不支持的数据库，可以使用 Connect()函数代替 PConnect()函数。

ADODB 支持多种数据库的连接，下面简单介绍几种常用数据库的连接方法。

1. MySQL

连接 MySQL 数据库有两种不同的方法。

第一种使用标准的连接字符串来连接。

代码如下：

```
<?php
    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $conn = ADONewConnection('mysql');                           //建立连接
    $conn -> PConnect('localhost','root','111','db_database19');  //连接数据库
?>
```

其中，localhost、root、111 和 db_database19 分别表示要连接数据库服务器的地址、用户名、密码和数据库名称。

第二种是通过数据源名称（DSN）的方式进行连接。

代码如下：

```
<?php
    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $conn = ADONewConnection('mysql://root:111@localhost/db_database19');   //连接数据库
?>
```

其实现的功能是相同的。

2. Microsoft SQL Server

连接 Microsoft SQL Server 数据库使用 ODBC 的连接方法。

代码如下：

```
<?php
    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $conn = ADONewConnection('odbc_mssql');                      //连接 SQL 数据库
    $conn-> PConnect("Driver={SQL Server};Server=localhost;Database=mydb;",'username','password');
?>
```

其中，localhost、username、password 和 mydb 分别表示要连接数据库服务器的地址、用户名、密码和数据库名称。

例 19.2 通过 ADODB 使用 ODBC 方法连接 Microsoft SQL Server 数据库，查询 Microsoft SQL Server 系统库 master 中的表 systypes，查询条件是 xtype 的值为 173，并输出查询结果。（**实例位置：光盘\TM\Instance\19\19.2**）

代码如下：

```
<?php

    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $conn = ADONewConnection('odbc_mssql');                      //连接 SQL 数据库
    $conn-> PConnect("Driver={SQL Server};Server=MR-NXT\NXT;Database=master;",'sa','');
    $ADODB_FETCH_MODE = ADODB_FETCH_ASSOC;                       //设置字段为结果集索引
    $sqlstr = 'select * from systypes where xtype = ?';          //创建 SQL 语句
    $rst = $conn -> execute($sqlstr,'173') or die('connect error');  //执行 SQL，查询 xtype 为 173 的数据
    while(!$rst -> EOF){                                         //如果没有错误，则配合 while 循环输出结果
        echo $rst -> fields['name'].'=>'.$rst -> fields['xtype'].'';  //输出查询结果
        $rst -> movenext();                                      //指针下移
    }
    $rst -> close();                                             //关闭连接
    $conn -> close();
?>
```

结果为：binary=>173。



3．Microsoft Access

连接 Access 数据库使用的是 ODBC 方法。

代码如下：

```
<?php
    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $conn = ADONewConnection('access');                          //连接 Access 数据库
    $conn-> PConnect("Driver={Microsoft Access Driver (*.mdb)};Dbq=d:\\mydb.mdb;Uid=Admin;Pwd=;");
?>
```

其中，d:\\mydb.mdb 表示数据库所在的文件名。

19.4.2　操作数据库函数

视频讲解：光盘\TM\Video\第 19 章\操作数据库函数.exe

1．execute($sql[,$inputarr=false])

函数作用：执行 SQL 语句，并返回一个结果集（ADORecordSet 对象）；如果失败则返回 false。

> **注意**　无论是执行 SELECT 语句，还是 INSERT 或 UPDATE 语句，只要执行成功，execute()函数都会返回一个结果集。

参数说明如下。

☑　$sql：要执行的 SQL 语句。

☑　$inputarr：用来作为传入的结合变量（Binding variables）。

没有变量$inputarr 时，$sql 为普通的 SQL 语句。execute()函数的格式为：

```
$connect->execute('select * from tb_object where id = 1');
```

当使用变量$inputarr 时，execute()函数的格式为：

```
$connect -> execute('select * from tb_object where id = ?',array($vl));
```

结合变量可以加速 SQL 指令编译及读取的速度，产生较佳的效能。

例 19.3　应用 execute()函数的第二个参数$inputarr，为 WHERE 子句补充上完整的条件，查询出数据表中 ID 等于 2 的记录。**（实例位置：光盘\TM\Instance\19\19.3）**

代码如下：

```
<?php
    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $ADODB_FETCH_MODE = ADODB_FETCH_ASSOC;                       //设置字段为结果集索引
    $conn = ADONewConnection('mysql');                           //建立连接
    $conn -> PConnect('localhost','root','111','db_database19'); //连接数据库
    $conn -> execute('set names gb2312');                        //设置编码格式
    $sqlstr = 'select * from tb_object where id = ?';            //创建 SQL 语句
    $rst = $conn -> execute($sqlstr,'2') or die('connect error'); //执行 SQL 语句，查询出 id 等于 2 的数据
    while(!$rst -> EOF){                                         //while 语句循环输出结果
        echo $rst -> fields['bigclass'].'=>'.$rst -> fields['d_time'].' '; //输出查询结果
        $rst -> movenext();                                      //指针下移
    }
    $rst -> close();                                             //关闭连接
    $conn -> close();
?>
```

结果为：图书开发=>2012-11-13 16:46:48。

2. cacheExecute([$sec,]$sql[,$inputarr=false])

函数作用：除了具有 execute()函数的功能外，还可将查询结果保存到缓存中。如果以后还有相同的查询，即可直接从缓存中获取。

参数说明如下。

- ☑ $sec：查询结果集在缓存中被保存的时间。
- ☑ $sql：执行的 select 查询语句。这里只能是 select 语句。
- ☑ $inputarr：结合变量。

3. selectLimit($sql[,$numrows=-1][,$offset=-1][,$inputarr=false])

函数作用：返回一个指定起始位置和记录数的结果集。MySQL 数据库支持 limit 查询功能，但 Access 等数据库系统并不支持 limit。ADODB 考虑到了这一点，模拟了一个和 limit 功能类似的函数。这里参数的位置和 MySQL 数据库中 limit 的参数位置不太一样。使用时需要注意。

参数说明如下。

- ☑ $sql：要执行的 select 查询语句。
- ☑ $numrows：是要查询的记录数。如果该值为-1，则函数会一直查询到最后一条记录。
- ☑ $offset：表示从第几条记录开始计算。
- ☑ $inputarr：结合变量。

例 19.4 使用 selectlimit()函数查询数据，从第$offset 条记录开始查起，返回$numrows 条记录，最后输出数据。**（实例位置：光盘\TM\Instance\19\19.4）**

代码如下：

```
<?php
	include_once 'conn/conn.php';                                      //载入数据库连接文件
	$sql = 'select * from tb_object';                                  //要查询的 SQL 语句
	$numrows = 2;                                                      //查找两条记录数
	$offset = 1;                                                       //从第一条记录开始计算
	$rst = $conn -> selectlimit($sql,$numrows,$offset) or die('execute error');   //调用 selectlimit()函数
	while(!$rst -> EOF){                                               //如果不是最后一条记录，循环
		echo $rst -> fields['id'].'=>'.$rst -> fields['bigclass'];     //输出 id 和字段值
		echo '<br>';
		$rst -> movenext();                                            //指针移动一位
	}
?>
```

结果为：网络=>2012-11-19 11:16:32 电器=>2012-11-25 14:56:21。

4. CacheSelectLimit([$sec,] $sql[, $numrows=-1][,$offset=-1][,$inputarr=false])

函数作用：除了具有 selectlimit()函数的功能外，还将查询结果保存到缓存当中。如果以后有相同的查询，将直接从缓存中获取。

参数含义同上。

5. CachFlush()

函数作用：清除所有 ADODB 数据库的缓存。

6. getUpdateSQL(&$rs, $arrFields, $forceUpdate=false)

函数作用：更新记录信息。

参数说明如下。

- ☑ $rs：要更新的结果集。



☑　$arrFields：更新字段及内容。

☑　$forceUpdate：如果$forceUpdate 为 true，无论$arrFields 和$rs 的值是否相等都将更新。

例 19.5　本例中首先查找 ID 等于 2 的记录，然后创建要更新字段的数组，最后将该数组内容通过 getUpdateSQL()函数更新到数据库中。（**实例位置：光盘\TM\Instance\19\19.5**）

代码如下：

```
<?php
    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $conn = ADONewConnection('mysql');                           //建立连接
    $conn -> PConnect('localhost','root','root','db_database19');    //连接数据库
    $conn -> execute('set names gb2312');                        //设置编码格式
    $sqlstr = 'select * from tb_object where id = 2';            //声明 SQL 语句
    $rst = $conn -> execute($sqlstr) or die('Error: '.$conn -> errorMsg());      //执行 SQL 语句
    echo '修改前的数据，时间为：'.$rst -> fields['d_time'].'<br>';         //输出数据库中的时间
    $fields = array();                                           //声明一个空数组$fields
    $fields['d_time'] = date("Y-m-d H:i:s");                     //将当前时间存到数组中
    $update = $conn -> getUpdateSQL($rst,$fields) or die('update error：'.$conn -> errorMsg());//更新数据
    $conn -> execute($update);                                   //执行更新
    echo '修改后的数据，时间为：';
    $rst = $conn -> execute($sqlstr);                            //执行查询
    echo $rst -> fields['d_time'];                               //输出更新后的信息
    $rst -> close();                                             //关闭连接
    $conn -> close();
?>
```

结果为：更新前的时间：2012-11-19 11:16:32　　更新后的时间：2012-11-24 07:31:44。

7．getInsertSQL(&$rs, $arrFields)

函数作用：添加新记录。

参数说明如下。

☑　$rs：要添加记录的结果集。

☑　$arrFields：新记录的字段值及对应字段名。

例 19.6　使用 getInsertSQL()函数向表 tb_object 中插入一条新数据。（**实例位置：光盘\TM\Instance\19\19.6**）

实例代码如下：

```
<?php
    include_once ('../adodb5/adodb.inc.php');                    //载入（include）adodb.inc.php 文件
    $ADODB_FETCH_MODE = ADODB_FETCH_ASSOC;                       //设置结果集按照字段名称为索引进行存取
    $conn = ADONewConnection('mysql');                           //建立连接
    $conn -> PConnect('localhost','root','111','db_database19');     //连接数据库
    $conn -> execute('set names gb2312');                        //设置编码格式
    $sqlstr = 'select * from tb_object where id = 0';            //生成 SQL 语句
    $rst = $conn -> execute($sqlstr) or die('Error: '.$conn -> errorMsg()) ;      //执行 SQL 语句
    $fields = array();                                           //创建一个数组
    $fields['bigclass'] = '办公用品';                              //向数组中添加数据
    $fields['d_time'] = date("Y-m-d H:i:s");
    $insert = $conn -> getInsertSQL($rst,$fields) or die('update error：'.$conn -> errorMsg()); //添加新数据
    $conn -> execute($insert);                                   //执行添加
    echo '添加的数据为：';
    $showsql = "select * from tb_object where d_time = '".$fields['d_time']."'";      //创建 SQL 语句
    $rst = $conn -> execute($showsql);                           //查找刚创建的记录
    foreach($rst -> fields as $nm => $vl)                        //使用 foreach 输出新记录
        echo $nm.'=>'.$vl.' ';
```

```
    $rst -> close();                                          //关闭连接
    $conn -> close();
?>
```

结果为：添加的数据为：id=>78 bigclass=>办公用品 d_time=>2012-11-24 07:38:50。

8．DBDate($date)

函数作用：转换不同数据库之间的日期格式。例如，MySQL 使用的时间格式为 Y-m-d，Oracle 则为 D-M-Y。

参数$date 是要存储的时间。

9．qstr($string)

函数作用：使用引号来处理字符串。例如，一个字符串包含了单引号（'），在 MySQL 中可以直接使用“'”表示，但在其他的数据库中，则使用两个单引号（''）表示。使用 qstr()函数可以处理和解决这个问题。

参数$string 为要被处理的字符串。

例 19.7 使用 DBDate()和 qstr()函数向表 tb_object 中插入一条数据。**（实例位置：光盘\TM\Instance\19\19.7）**

代码如下：

```
<?php
    include_once 'conn/conn.php';                             //载入数据库连接文件
    $bigclass = $conn -> qstr("建筑耗材");                     //用 qstr()函数处理带引号的字串
    $d_time = $conn -> DBDate(date('Y-m-d H:i:s'));           //用 DBDate()函数处理日期时间
    $sql = 'insert into tb_object(bigclass,d_time) value('.$bigclass.','.$d_time.')';    //生成 insert 语句
    $rst = $conn -> execute($sql) or die($conn -> errorMsg());                            //执行 insert 语句
    if($rst)
        echo '添加成功';                                                                  //添加成功
?>
```

结果为：添加成功。

说明 在本实例中，为了使代码整洁，便于对使用数据库的管理，将连接数据库的操作定义到 conn.php 文件中，在使用时只要通过 include_once 语句进行调用即可。

10．Affected_rows()

函数作用：返回最后更新或被删除的记录数。如果数据库不支持，则返回 false。

例 19.8 对 id 为 2 的记录进行修改，将该记录的 d_time 字段值设为当前时间，然后调用 Affected_rows()函数查看返回结果。**（实例位置：光盘\TM\Instance\19\19.8）**

代码如下：

```
<?php
    include_once 'conn/conn.php';                                                    //载入数据库连接文件
    $sqlstr = 'update tb_object set d_time = now() where id = 2';                    //SQL 更新语句
    $rst = $conn -> execute($sqlstr) or die('update error: '.$conn -> errorMsg());   //执行更新语句
    echo '更新记录数：'.$conn -> Affected_rows();                                     //查看更新的记录数
?>
```

结果为：更新记录数：1。

11．Insert_ID()

函数作用：返回最后插入的记录 ID 值。如果数据库不支持该函数，则返回 false。



19.4.3　控制结果集存取方式

视频讲解：光盘\TM\Video\第 19 章\控制结果集存取方式.exe

1. $ADODB_COUNTRECS

当该变量为 true 时，RecordCount()函数会在数据库驱动不支持 select 指令返回的记录总数时进行自动模拟，返回记录数，默认值为 true。每次进行查询操作时都会自动检查该变量的值。由于这个功能十分消耗内存，所以不建议使用。

2. $ADODB_CACHE_DIR

该变量用于设置缓存目录。一般在使用缓存函数时会用到该变量，如 CacheExecute()函数。

3. $ADODB_FETCH_MODE

该变量决定结果集以哪种方式进行存取。其有 4 个值：

（1）define('ADODB_FETCH_DEFAULT',0)

（2）define('ADODB_FETCH_NUM',1)

结果集按照字段序号为索引进行存取，如$rst -> fields[0]、$rst -> fields[1]。

（3）define('ADODB_FETCH_ASSOC',2)

结果集按照字段名称为索引进行存取，如$rst -> fields[bigclass]、$rst -> fields[id]。

（4）define('ADODB_FETCH_BOTH',3)

结果集同时支持（2）、（3）两种方式。如果未设置该变量，默认值为（1），不同类型的数据库驱动默认值也不相同。这里不建议使用变量（4），因为有很多驱动并不支持。

例 19.9　仍使用例 19.1 中的内容，只是增加了一个公共变量，将结果集以字段名作为索引进行输出。**（实例位置：光盘\TM\Instance\19\19.9）**

代码如下：

```
<?php
	include_once ('../adodb5/adodb.inc.php');				//载入（include）adodb.inc.php 文件
	$ADODB_FETCH_MODE = ADODB_FETCH_ASSOC;				//设置公共变量值
	$conn = ADONewConnection('mysql');					//建立连接 MySQL 数据库
	Connect($host,[$user],[$password],[$database])			//连接数据库
	$conn -> PConnect('localhost','root','111','db_database19');		//连接指定的数据库
	$conn -> execute('set names gb2312');				//设置编码格式
	$rst = $conn -> Execute('select * from tb_object') or die('执行错误');	//执行查询语句
	while(!$rst -> EOF){							//while 语句循环输出结果
		echo $rst -> fields['id'].'=>'.$rst -> fields['bigclass'].' ';	//循环输出记录
		$rst -> movenext();						//指针下移
	}
	$rst -> close();							//关闭连接方法
	$conn -> close();							//关闭数据库
?>
```

结果为：1=>图书开发 2=>网络 3=>电器 78=>办公用品 79=>网络。

19.4.4　操作结果集函数

视频讲解：光盘\TM\Video\第 19 章\操作结果集函数.exe

当使用 execute()函数执行 SQL 指令时，会回传一个结果集。该结果集实际上是一个 ADORecordSet 对

象。通过对该对象的控制，可以对结果进行各项操作。

1. GetArray([$number_of_rows])

函数作用：返回从当前指针指向的记录开始，到（$number_of_rows–1）行的全部记录的数组。

参数$number_of_rows 表示指定的记录行。如果没有给出，则一直到 EOF 才停止。

2. UserDate($str, [$fmt])

函数作用：将日期字符串$str 转换为变量$fmt 设置的日期格式。

参数说明如下。

☑ $str：日期格式的字符串。

☑ $fmt：要转换的日期格式，默认值为 Y-m-d。

3. UserTimeStamp($str, [$fmt])

函数作用：将时间字符串$str 转换为变量$fmt 设置的时间格式。

参数说明如下。

☑ $str：时间字符串。

☑ $fmt：时间格式设置字串，默认值为 Y-m-d H:i:s。

4. MoveNext()

函数作用：将 ADORecordSet（结果集）的指针下移一位。如果成功，则返回 true，否则返回 false。

5. Move($to)

函数作用：将 ADORecordSet（结果集）的指针移动到指定位置。如果$to 等于 0，则指针指向结果集的第一条数据；如果$to 的值大于结果集，则指针指向最后一条数据。注意，这里的变量$to 只能是绝对定位。

6. MoveFirst()

函数作用：指针移动到第一条数据，等同于 Move(0)。

7. MoveLast()

函数作用：指针移动到最后一条数据，等同于 Move(RecordCount()−1)。

8. FetchRow()

函数作用：返回当前指针指向的记录的数组，如果是 EOF，则回传 false。FetchRow()不要和 MoveNext()混用。

9. FetchField($column_number)

函数作用：返回一个对象，包含字段$column_number 的名称、空间长度等相关信息。

参数$column_number 表示要查看的字段名称。

10. FetchNextObject([$toupper=true])

函数作用：返回当前指针所指向的记录的对象形式，并且指针自动下移一行。

参数$toupper：如果等于 true，则字段名为大写形式。

11. FieldCount()

函数作用：返回结果集中的字段数。

12. RecordCount()

函数作用：返回结果集中的记录数。



13．CurrentRow()

函数作用：返回当前指针所指的记录序号。第一条记录用 0 表示。

14．MetaType($nativeDBType [,$field_max_length],[$fieldobj])

函数作用：标准化不同数据库系统下的数据类型的表示法。处理不同的数据库系统有一个问题：每一个数据库系统对于相同的数据类型会有不同的表示法，如支持超长文本的类型，有的表示为 text，有的表示为 long character，还有 clob 等。MetaType()函数可以将这几种表示法统一起来，如用大写字母 D 来表示所有的日期类型，用大写字母 T 表示所有的时间类型。

参数说明如下。

☑　$nativeDBType：数据表中的数据类型。
☑　$fields_max_length：字段的最大长度。
☑　$fieldobj：字段对象。

MetaType()函数支持的标准化数据类型如下。

☑　C：char、varchar 等字符类型。
☑　X：text、long character 等长类型。
☑　B：blob 或 binary image。
☑　D：日期 date。
☑　T：时间 timestamp。
☑　L：逻辑类型和位类型。
☑　N：包含编号、整型、浮点型等数字类型。
☑　R：包含序列、自动增加整数等序列类型。

例 19.10　通过 FieldCount()函数获取结果集记录数，然后通过 for 循环输出数据，并对每个字段类型进行判断，如果类型为 T，则使用 DBDate()函数格式化输出时间，其他类型则直接输出。（**实例位置：光盘\TM\Instance\19\19.10**）

代码如下：

```
<?php
	include_once 'conn/conn.php';                                         //载入数据库连接文件
	$sqlstr = 'select * from tb_object where id = 1';                      //SQL 查询语句
	$rst = $conn -> execute($sqlstr) or die('error：  '.$conn -> errorMsg());  //执行查询语句
	if(false != $rst){                                                    //如果有查询结果
		for($i = 0; $i < $rst -> FieldCount(); $i++){                     //循环输出各个字段
			$fields = $rst -> FetchField($i);                             //生成字段信息对象
			$type = $fields -> type;                                      //从对象中获取字段的类型信息
			echo '=>';
			echo $rst -> metaType($type,-1,$fields);                      //输出字段标准类型
			if($rst -> metaType($type,-1,$fields) == "T")                 //如果标准类型为 T
				echo '('.$conn -> DBDate($rst -> fields[$i]).')';         //使用 DBDate 函数格式化时间
			else
				echo '('.$rst -> fields[$i].')';                          //如果是其他类型，直接输出
		}
	}
?>
```

结果为：=>R(1)=>C(图书开发)=>T('2012-11-13 16:46:48')。

19.4.5　处理事务函数

1．BeginTrans()

函数作用：开启事务处理。该函数和下面介绍的两个函数都应用在 PHP 的事务处理中。

2．CommitTrans()

函数作用：如果成功完成数据库操作，则执行 CommitTrans()函数。

3．RollbackTrans()

函数作用：结束一次操作，恢复操作前的所有改变。如果成功则返回 true，如果服务器不支持则返回 false。

19.4.6　生成 HTML 表格函数

rs2html($rst,[$table_attributes], [$col_titles])

函数作用：返回一个 HTML 表格格式的结果集。要使用该函数，需要载入 tohtml.inc.php 文件。

参数说明如下。

☑　$rst：要返回的结果集。

☑　$table_attributes：对表格参数及属性的设置。

☑　$col_titles：对字段的重新命名。

例 19.11　应用 rs2html()函数直接输出返回的结果集$rst，并设置表格的属性。**（实例位置：光盘\TM\Instance\19\19.11）**

代码如下：

```
<?php
	include_once 'conn/conn.php';                                   //载入数据库连接文件
	include_once '../adodb5/tohtml.inc.php';                        //载入 tohtml.inc.php 文件
	$rst = $conn -> execute('select * from tb_object');             //返回查询结果集
	rs2html($rst,' width="350" border="1" cellpadding="1" cellspacing="1" bordercolor="#FFFFFF" bgcolor="#FF0000"',
array('序号','类别','添加时间'));                                    //输出结果集
?>
```

运行结果如图 19.1 所示。

图 19.1　应用 rs2html()函数直接输出返回的结果集

19.4.7　生成下拉列表框函数

视频讲解：光盘\TM\Video\第 19 章\生成下拉列表框函数.exe

1. getMenu($name, [$default_str = ''], [$blank = true], [$multiple_select = false], [$size = 0], [$moreAttr = ''])

函数作用：返回一个 HTML 下拉列表框（select）。结果集的第一列（fields[0]）被显示到下拉列表中，第二列（fields[1]）被设置为<option>标签的 value 属性。

参数说明如下。

☑　$name：下拉列表框的名称。

☑　$default_str：如果$default_str 的值等于其中一个 fields[0]，那么这个 fields[0]就是该下拉列表的默

认选项。这相当于为 fileds[0]添加了属性<selected>。

☑ $blank：该参数的默认值为 true。在生成下拉列表框时，第一行选项默认为空，即<option></option>。如果为 false，则没有默认的空选项。

☑ $multiple_select：是否支持多选。默认值为 false，即不允许多选。

☑ $size：列表框选取的大小。默认为 0，生成一个下拉列表；如果$size 的值大于 0，则生成一个列表框。

☑ $moreAttr：该参数可以添加列表框的属性，如添加一个 JavaScript 脚本等。

2. getMenu2($name, [$default_str=''], [$blank1stItem=true], [$multiple_select=false], [$size=0], [$moreAttr=''])

函数作用和参数含义同上。该函数和 GetMenu()函数的不同之处是：getMenu()函数中的参数$default_str 和 fields[0]比较，如果相等，那么 fields[0]就是下拉列表的默认选项；而 getMenu2()函数中的参数$default_str 则是和 fields[1]进行比较，其他不变。

例 19.12　本例中首先使用查询语句和 execute()函数返回一个结果集，然后在一个 form 表单中使用 getMenu2()函数将结果集输出到下拉列表框中。当单击"提交"按钮时，通过$_POST[]预定义变量获取下拉列表框提交的值。**（实例位置：光盘\TM\Instance\19\19.12）**

代码如下：

```
<?php
include_once 'conn/conn.php';                                         //载入数据库连接文件
$sqlstr = 'select bigclass,id from tb_object';                        //SQL 查询语句
$rst = $conn -> execute($sqlstr) or die('execute error:  '.$conn -> errorMsg());   //执行查询语句，返回结果集
?>
<!--  提交表单  -->
<form method='post' action=''>
    请选择购买商品的类别：
<!--  调用 getMenu2()函数  -->
    <?php echo $rst -> getMenu2('bigclass',$_POST[bigclass],false); ?>
    <input type='submit' name='submit' value='提交' />
</form>
<!--  ---------------  -->
<?php
if(isset($_POST[bigclass])){                                          //判断表单是否被提交
    echo '您选择的商品类别是：'.$_POST[bigclass];                      //输出提交的表单值
}
?>
```

运行结果如图 19.2 所示。

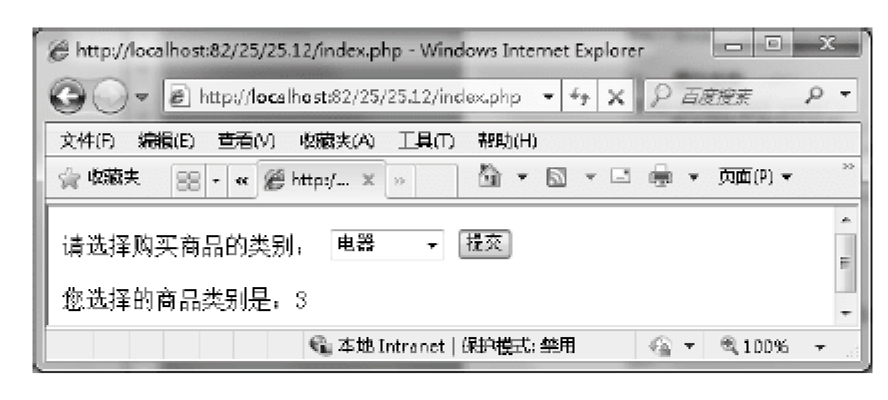


图 19.2　使用 getMenu2()函数输出下拉列表框

19.4.8　实现分页功能函数

视频讲解：光盘\TM\Video\第 19 章\实现分页功能函数.exe

1. PageExecute($sql, $nrows, $page)

函数作用：分页功能函数。

参数说明如下。

☑ $sql：SQL 查询语句。

☑ $nrows：每页显示的记录数。

☑ $page：保存当前页数，默认为 1。

2．CachePageExecute($sec, $sql, $nrows, $page)

函数作用：分页显示函数，同时将显示后的记录放到缓存中。在$sec 秒内，如果是重复查看，将从缓存中读取数据。

参数$sec 表示数据在缓存中保留的时间。其他参数见 PageExecute()函数。

3．AbsolutePage($page=-1)

函数作用：返回当前的页数。要和 PageExecute()函数配合使用，用于分页。

4．AtFirstPage($status='')

函数作用：如果当前是第一页，返回 true。要和 PageExecute()函数配合使用，用于分页。

5．AtLastPage($status='')

函数作用：如果当前是最后一页，返回 true。要和 PageExecute()函数配合使用，用于分页。

6．ADODB_Pager($conn, $sql, $id = 'adodb', $showPageLinks = false)

函数说明：该函数是 adodb_pager 类的构造函数，通过类中的 render()函数可以实现一个小巧的分页操作。如果要使用这个功能，需要载入 adodb_paper.inc.php 页。

参数说明如下。

- ☑ $conn：数据库连接对象。
- ☑ $sql：执行的 SQL 语句。
- ☑ $id：每个分页的 id 号。
- ☑ $showPageLinks：是否显示各个页的链接，默认为 false。

例 19.13 使用 adodb_pager 类将表 tb_object 中的记录分页显示出来。在本实例中对载入文件 adodb_pager.inc.php 进行了修改。**（实例位置：光盘\TM\Instance\19\19.13）**

修改的代码如下：

```
    var $first = '<code>首页</code>';
    var $prev = '<code>上一页</code>';
    var $next = '<code>下一页</code>';
    var $last = '<code>尾页</code>';
```

上述修改是为了将翻页改成中文连接。同样，在应用的过程中还可以修改 adodb_pager.inc.php 文件中的其他内容，如表格的背景、宽度，以及边框的宽度、颜色等。

本实例的代码如下：

```
<?php
    include 'conn/conn.php';                                    //载入数据库连接文件
    include_once '../adodb5/adodb_pager.inc.php';               //载入 adodb_pager.inc.php 文件
    /*   声明对象   */
    $pager = new ADODB_Pager($conn,"select id,bigclass as '大类', d_time as '添加时间' from tb_object");
    /*   调用$Render 函数确定每页显示多少   */
    $pager -> Render(3);
?>
```

运行结果如图 19.3 所示。

19.4.9 错误处理函数

ADODB 的优势是可以让各种数据库函数统一起来，对相同的数据进行相同的操作。但在实际操作时，难免会出现各种问题。下面介绍几种错误处理及调试的方法。



1．debug

如果变量被设定为 true，当有输出操作时，同时也输出调试信息。

2．errorMsg()

函数作用：返回最后的状态或出错信息。即使没有错误发生，也会返回一个字串。所以，一般情况下，只有在发生错误时（返回 false）才调用该函数。在启用 debug 变量后，只要有错误发生就会自动调用该函数。

例 19.14　人为地造成一个错误，将 SQL 语句中的 from 写成 form，来看一下输出的错误信息。**（实例位置：光盘\TM\Instance\19\19.14）**

代码如下：

```
<?php
    include_once 'conn/conn.php';                               //载入数据库连接文件
    $conn -> debug = true;                                      //开启调试
    $sql = 'select * form tb_object';                           //SQL 查询语句
    $rst = $conn -> execute($sql) or die($conn -> errorMsg());  //调用查询语句
?>
```

运行结果如图 19.4 所示。





图 19.3　adodb_pager 类用于显示记录分页

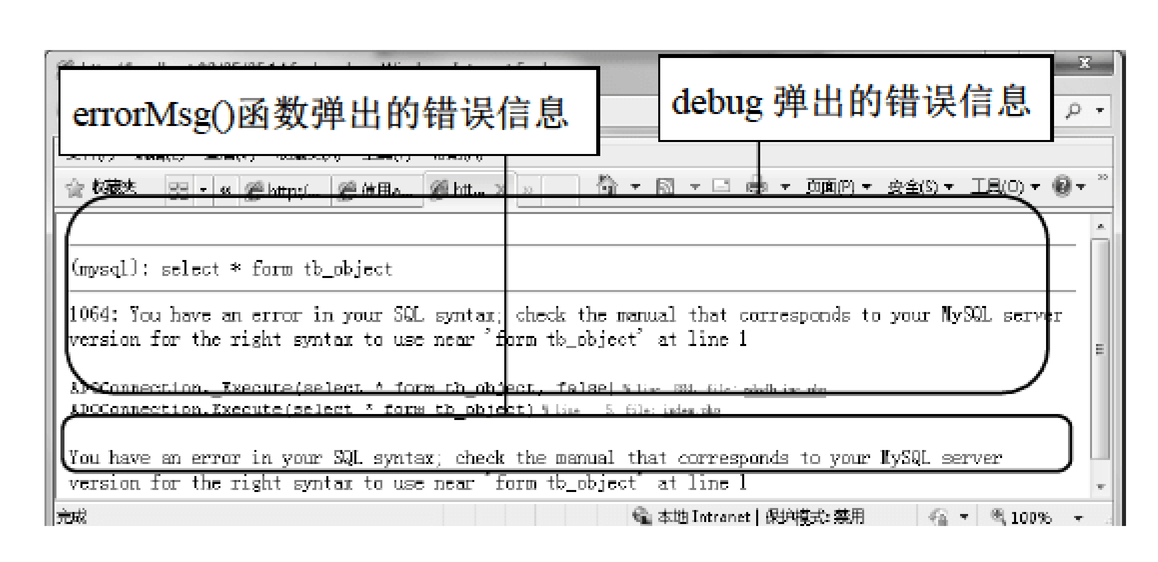


图 19.4　ADODB 调试

可以发现，debug 弹出的错误提示中，已经包含了 errorMsg()函数的错误信息。

19.5　实　　战

视频讲解：光盘\TM\Video\第 19 章\实战.exe

19.5.1　实现分页

例 19.15　除了可以使用 adodb_pager.inc.php 中的类实现分页效果外，用 ADODB 类库中默认的几个函数同样可以实现分页效果。分页主要使用 PageExecute()、AbsolutePage()、AtFirstPage()、AtLastPage()和 LastPageNo() 5 个函数来完成。**（实例位置：光盘\TM\Instance\19\19.15）**

实例代码如下：

```
<table width="384" border="0" cellpadding="0" cellspacing="0">
    <tr><td height="30">
<?php
    include_once 'conn/conn.php';                   //载入数据库连接文件
    include('../adodb5/tohtml.inc.php');            //载入 tohtml.inc.php 文件
    $sql = 'select * from tb_object';               //查询语句
```

```
        $num = 2;                                              //每页显示的记录数
        if(isset($_GET['n_page'])){                            //判断当前页码
                $c_page = $_GET[n_page];                       //将$n_page 赋给变量$c_page
        }else{
                $c_page = 1;                                   //初始化变量$c_page
        }
        $rst = $conn -> PageExecute($sql,$num,$c_page);        //执行 PageExecute()函数
        if(false != $rst){
        if(!$rst -> AtfirstPage()){                            //如果当前页不是首页
?>              <!-- 输出向上翻页超链接 -->
                <a href ="<?php echo '?n_pge=1' ?>"> 首页 </a>
                <a href ="<?php echo '?n_page='.($rst -> AbsolutePage() - 1); ?>"> 上一页 </a>
                <!-- --------------------------- -->
<?php
        }
        if(!$rst -> AtlastPage()){                             //如果当前页不是尾页
?>
                <!-- 输出向下翻页超链接 -->
                <a href = "<?php echo '?n_page='.($rst -> AbsolutePage() + 1); ?>"> 下一页 </a>
                <a href ="<?php echo '?n_page='.($rst -> LastPageNo());?>"> 尾页 </a>
                <!-- ---------------------------- -->
<?php
        }
?>
        </td></tr>
        <tr><td>
<?php
        rs2html($rst,'width=350 border="1" bgcolor="#FF0000"',array('ID','类型','添加时间'));
?>
        </td></tr>
<?php     }     ?>
        <tr><td height="30">
                当前是第<?php echo $rst -> AbsolutePage(); ?>页/一共是<?php echo $rst -> LastPageNo(); ?>页
</td></tr>
</table>
```

运行结果如图 19.5 所示。

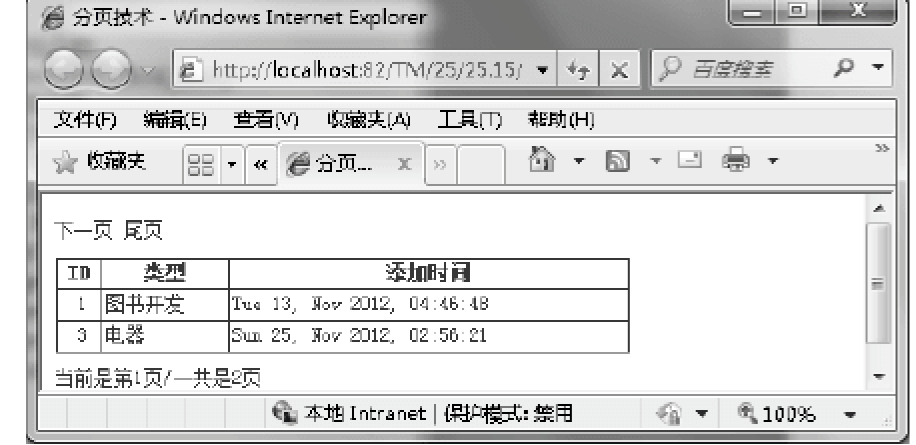


图 19.5 分页技术

19.5.2 处理事务

在对数据库进行操作时，由于发生意外，导致数据更新到一半时就停止了。这时可以通过事务将数据恢复到更新以前状态。下面就来看一个通过事务实现回滚的实例。

例 19.16 在 ADODB 类库中，通过 BeginTrans()、CommitTrans()和 RollbackTrans() 3 个函数实现处理事务。**（实例位置：光盘\TM\Instance\19\19.16）**

实例代码如下：

```
<?php
        include_once 'conn/conn.php';                          //载入数据库连接文件
        $conn -> BeginTrans();                                 //开始事务处理
        $sql = 'delete from tb_object where id = 2';           //SQL 删除语句
        $rst = $conn -> execute($sql) or die('execute error: '.$conn -> ErrorMsg());    //执行删除语句
        $num = $conn -> Affected_rows();                       //查看被更新的记录数
        if(false !== $rst){                                    //如果$rst 不为假
```

```
            if($num != 0){                                      //如果$num 不为 0，说明删除成功
                $conn -> CommitTrans();                         //执行提交
                echo '删除成功！';
                exit();
            }else{                                              //如果$num 为 0，说明没有删除记录
                echo '没有数据，或数据已删除';
                exit();
            }
        }else{                                                  //如果发生意外
            $conn -> RollbackTrans();                           //执行回滚操作
            echo '出现意外。';
        }
?>
```

本实例如果存在指定的数据，则执行删除后返回“删除成功！”；如果不存在，则输出“没有数据，或数据已经删除”；如果出现意外，则执行回滚并输出“出现意外。”。

19.5.3　缓存函数+ADODB 动态生成静态页

现在很多站点都采用动态生成静态页面的方式，这样不仅可以有效地提高站点的访问效率，降低服务器负载，而且可以提高被搜索引擎搜索到的几率。

例 19.17　使用缓存函数和 ADODB 类库制作能够动态生成静态页面的学生成绩管理系统。（**实例位置：光盘\TM\Instance\19\19.17**）

实现步骤如下。

（1）本实例使用 ADODB 技术管理 MySQL 数据库，为了提高代码重用率，体现 PHP 面向对象编程的特点，开发实例时创建数据库连接类建立与 MySQL 数据库的连接。

代码如下：

```
<?php
class ConnDB
{
    var $dbType;                                                //数据库类型
    var $host;                                                  //MySQL 数据库服务器地址
    var $userName;                                              //用户名
    var $password;                                              //密码
    var $dbName;                                                //数据库名
    var $isDebug;                                               //是否调试
    var $connID;                                                //保存数据库连接标识

    function ConnDB ($dbType = 'mysql', $host, $userName, $password, $dbName, $isDebug = false)
    {
        //构造函数，用于对类中的属性进行初始化
        $this->dbType = $dbType;
        $this->host = $host;
        $this->userName = $userName;
        $this->password = $password;
        $this->dbName = $dbName;
        $this->isDebug = $isDebug;
    }

    function getConnID ()
    {
        require_once 'library/adodb/adodb.inc.php';             //导入 ADODB 类库
```

```
        $this->connID = NewADOConnection($this->dbType);        //建立连接
        if ($this->dbType == 'mysql' || $this->dbType == 'mssql') {  //MySQL 或 SQL Server 数据库
            $this->connID->Connect($this->host, $this->userName, $this->password, $this->dbName);
            if ($this->dbType == 'mysql') {
                $this->connID->Execute('set names gbk');        //如果是 MySQL 数据库则设置字符集
            }
        } elseif ($this->dbType == 'access') {                  //Access 数据库
            $this->connID->Connect('Driver = {Microsoft Access Driver (*.mdb)}; Dbq =' . realpath($this->
dbName) . ';Uid = ' . $this->userName . ';Pwd = ' . $this->password. ';');
        } else {
            return false;
        }
        $this->connID->debug = $this->isDebug;                  //是否调试
        return $this->connID;
    }

    function closeConnID ()                                     //关闭与数据库的连接
    {
        @$this->connID->Disconnect();
    }
}
```

上述代码主要应用 ConnDB 类通过 ADODB 类库建立与 MySQL 数据库的连接。分析代码可知，ConnDB 类不仅可以建立与 MySQL 数据库的连接，而且可以建立与 Access 和 SQL Server 数据库的连接。ConnDB 类中首先使用构造方法对类中的属性进行实例化，然后创建 getConnID()方法用于返回数据库连接的 ID，最后创建 closeConnID()方法释放数据库连接资源。

（2）建立基于 ADODB 类库的分页类，用于实现学生成绩信息的分页显示。

代码如下：

```
<?php
class PageDB
{
    function pageData ($sql, $connID, $pageSize, $page)
    {
        $arrayPageInfo = array();                       //定义要返回的数组
        if (isset($page) && $page != '') {              //判断是否有页码传递
            $nowPage = $_GET['page'];                   //有则获取页码的值
        } else {
            $nowPage = 1;                               //没有则默认显示第一页
        }
        $rs = $connID->PageExecute($sql, $pageSize, $nowPage);
        $arrayData = $rs->GetRows();                    //执行查询
        if (count($arrayData) == 0 || $rs == false) {
            return false;                               //如果出错或没有查询到则返回 false
        } else {
            $arrayPageInfo['data'] = $arrayData;        //查询到的数据
            $rsAll = $connID->Execute($sql);
            $countRs = count($rsAll->GetRows());        //总记录数
            $countPage = ceil($countRs / $pageSize);    //总页数
            $arrayPageInfo['pageSize'] = $pageSize;     //将分页信息保存到要返回的数组$arrayPageInfo 中
            $arrayPageInfo['countRs'] = $countRs;
            $arrayPageInfo['page'] = $nowPage;
            $arrayPageInfo['countPage'] = $countPage;
            $arrayPageInfo['first'] = 1;
            if ($page > 1) {
                $arrayPageInfo['previous'] = $rs->AbsolutePage() - 1;
            } else {
```

```
                    $arrayPageInfo['previous'] = 1;
                }
                if ($page < $countPage) {
                    $arrayPageInfo['next'] = $rs->AbsolutePage() + 1;
                } else {
                    $arrayPageInfo['next'] = $countPage;
                }
                $arrayPageInfo['last'] = $countPage;
            }
            return $arrayPageInfo;                          //返回包含所有分页信息的数组
        }
}
```

PageDB 类主要实现 ADODB 类库操作 MySQL 数据库的分页显示，类中所定义的 pageData()方法用于返回一个参数数组，该数组包含所有分页信息，如每页显示的记录数、总页数、总记录数及分页导航链接具体页码值等，这样只需根据返回数组即可获取所有分页信息，为进一步开发做好铺垫。

（3）建立 StaticPage 类用于动态生成静态页面，该类主要使用 PHP 中缓存函数实现。

具体实现代码如下：

```
class StaticPage
{
    function pageBegin ()                               //指定要生成静态页的开始部分
    {
        ob_start();
    }

    function pageEnd ()                                 //定义要生成静态页的结束部分
    {
        ob_end_flush();
    }

    function getStaticPageName ($url, $extendsName, $defaultPage = 'index')
    {
        //根据动态页的实际地址生成静态页的名称
        $baseName = basename($url);                                         //不包含目录的 PHP 动态页名
        $pageName = substr($baseName, 0, strpos($baseName, '.php'));       //去掉扩展名
        if ($pageName == '') {
            $pageName = $defaultPage;
        }
        $searchString = substr(strchr($baseName, '?'), 1, strlen(strchr($baseName, '?')));
                                                                            //获取查询字符串部分
        $arraySearchString = explode('&', $searchString);                   //分割查询字符串并保存到数组中
        $searchName = '';
        for ($i = 0; $i < count($arraySearchString); $i ++) {               //使用“-”连接查询字符串的值
            $searchStringTmpValue = strstr($arraySearchString[$i], '=');
            if ($searchStringTmpValue != '') {
                $searchName .= '-' . substr($searchStringTmpValue, 1, strlen($searchStringTmpValue));
            }
        }
        $fileName = $pageName . $searchName . '.' . $extendsName;           //生成静态页文件名
        return $fileName;                                                   //返回静态页文件名
    }

    function makeStaicPage ($url, $entendsName, $saveDir)
    {
        //生成静态页文件
        $content = ob_get_contents();                   //获取 PHP 页面生成的 HTML 页面内容
        if (! is_dir($saveDir)) {
```

```
            mkdir($saveDir);                              //创建保存静态页的目录
        }
        $fileName = $this->getStaticPageName($url, $entendsName);
        $fp = fopen($saveDir . '/' . $fileName, 'w');
        fwrite($fp, $content);                            //将 HTML 内容写入静态页
    }

    function reMakeCondition ($saveDirAndFile, $reMakeTime = -1)
    {
        //设定重新生成静态页的条件
        if ($reMakeTime == - 1) {
            if (file_exists($saveDirAndFile)) {
                return true;
            }
        } elseif (file_exists($saveDirAndFile) && time() - filemtime($saveDirAndFile) < $reMakeTime) {
            //如果静态页超出指定的时间范围或不存在则返回 true，表示将重新生成静态页
            return true;
        }
        return false;
    }
}
```

上述代码中定义的 pageBegin()和 pageEnd()方法分别用来定义所要生成的静态页面的开始和结束位置，定义的 getStaticPageName()方法用来根据实际 PHP 页面的请求地址生成静态页面地址，makeStaicPage()方法用于创建静态页面，最后定义的 reMakeCondition()方法用于指定重新生成静态页面的条件。

分析上述代码可知，本实例所讲述的生成静态页的方法是真正意义上的生成静态页面，而非伪静态，这样的好处是不受服务器局限，而使用伪静态的方法需要开启 Apache 服务器的 Rewrite 功能，如果用户使用租用的空间，可能有些空间提供商未提供上述技术支持。

（4）建立 index.php 页面，用于实现学生信息的分页显示，在页面最开始调用 StaticPage 类，并根据 reMakeCondition()方法判断与当前 PHP 页面所对应的静态页面是否存在或静态页面是否已经过期，如果是则重新生成，否则使用 header()函数重新定向到与该页面所对应的静态页面中，并使用 exit()函数终止该 PHP 页面剩余代码的执行，从而提高代码执行效率。

```
<?php
require_once 'StaticPage.php';                     //包含 StaticPage 类
$ststicPage = new StaticPage();                    //对 StaticPage 进行实例化
$ststicPage->pageBegin();                          //定义生成静态页面的起始位置
$dirName = './pages';                              //定义生成静态页的目录
$fileName = $ststicPage->getStaticPageName($_SERVER['REQUEST_URI'], 'shtml');
                                                   //生成与该 PHP 页面对应的静态页的名称
if ($ststicPage->reMakeCondition($dirName .'/'. $fileName, 60)){      //判断是否需要重新生成静态页
    header('location:pages/' . $fileName);         //不需要则重新定向到与该 PHP 页面所对应的静态页面
    exit();
}
$baseUrl = '/TM/19/19.17';
require_once 'ConnDB.php';                                     //包含数据库连接类
require_once 'PageDB.php';                                     //包含分页类
$connObj = new ConnDB('mysql', 'localhost', 'root', 'root', 'db_database19', false);    //实例数据库连接类
$connID = $connObj->getConnID();                               //返回数据库连接 ID
?>
```

（5）实例 PageDB 分页类，通过 for 循环显示学生信息。

```
<?php
    $pageObj = new PageDB();                                   //实例分页类
    $pageInfo = $pageObj->pageData("select id,studentno ,studentname,  classname,  tel , email from tb_
```

```
student order by id ", $connID, 5, $_GET['page']);            //获取包含分页信息的数组
      $arrayPageData = $pageInfo['data'];                      //分页数组下标为 data 项的元素为学生信息内容
      for ($i = 0; $i < count($arrayPageData); $i ++) {        //通过 for 循环显示学生信息
?>
<!--显示学生信息 -->
<?php
      }
?>
```

（6）显示分页链接，学生信息显示页面根据该链接参数 page 的值决定显示第几个页面。

```
<div style="width:80%; height:18px; text-align:left; padding-top:5px">
            共有学生信息 <?= $pageInfo['countRs']?> 条 每页显示 <?= $pageInfo
['pageSize'] ?>  条 第 <?= $pageInfo['page'] ?> 页/共 <?= $pageInfo['countPage']
?>
 页           
      <a href="<?= $_SERVER['PHP_SELF']?>?page=<?= $pageInfo['first']?>" class="a1">首页</a> 
 
      <a href="<?= $_SERVER['PHP_SELF']?>?page=<?= $pageInfo['previous']?>" class="a1">上一页</a> 
 
      <a href="<?= $_SERVER['PHP_SELF']?>?page=<?= $pageInfo['next']?>" class="a1">下一页</a> 
 
      <a href="<?= $_SERVER['PHP_SELF']?>?page=<?= $pageInfo['last']?>" class="a1">尾页</a>
</div>
```

（7）关闭数据库的连接，并调用 ssticPage 类的 makeStaicPage()方法生成静态页，最后调用 ssticPage 类的 pageEnd()方法指定生成静态页面的结束位置。

```
<?php
$connObj->closeConnID();                                             //关闭与数据库的连接
$ssticPage->makeStaicPage($_SERVER['REQUEST_URI'], 'shtml', './pages');       //生成静态页面
$ssticPage->pageEnd();                                               //指定生成静态页面的结束位置
?>
```

运行上述实例，当第一次访问学生信息页面时所浏览的是 PHP 动态页面的内容，如果返回页面继续浏览，将访问该 PHP 页面所对应的静态页，分别如图 19.6 和图 19.7 所示。

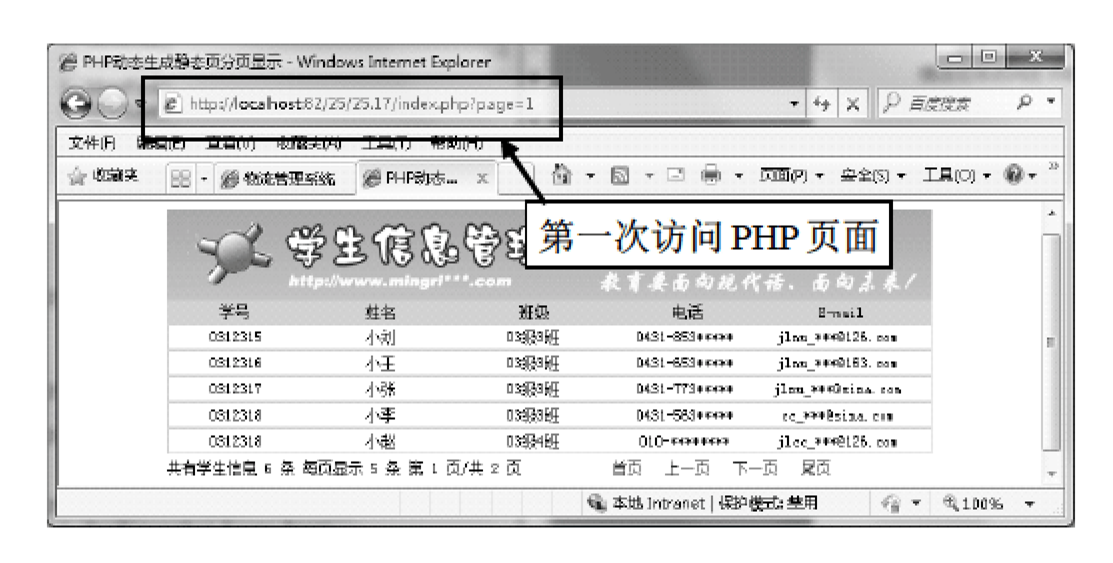


图 19.6　第一次访问 PHP 页面

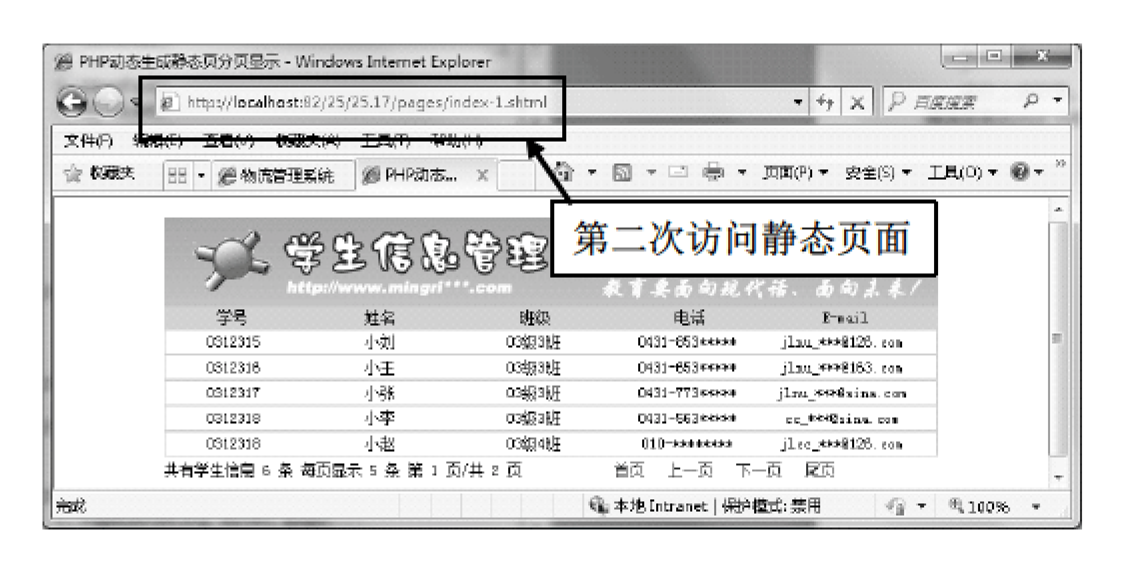


图 19.7　第二次及以后访问 PHP 页面

19.5.4　ADODB 控制结果集的存取方法

例 19.18　通过操作结果集方法，对数据表中的行进行操作，循环输出数据表中的数据。同时应用一个新的方法，实现对结果集存取方式的控制。（**实例位置：光盘\TM\Instance\19\19.18**）

实现步骤如下。

（1）创建 conn 文件夹，编写 conn.php 文件，载入 adodb.inc.php 文件。实现与 MySQL 数据库服务器中 db_database19 数据库的连接，应用$ADODB_FETCH_MODE 设置结果集以字段名称为索引进行存取，并

设置数据库的编码格式为 utf-8。

代码如下：

```
<?php
include_once ('../adodb5/adodb.inc.php');                    //载入 adodb.inc.php 文件
$conn = ADONewConnection('mysql');                           //连接 MySQL 数据库
$conn -> PConnect('localhost','root','111','db_database19'); //连接 db_database19 数据库
$ADODB_FETCH_MODE = ADODB_FETCH_ASSOC;                       //设置结果集以字段名称为索引
$conn -> execute('set names utf8');                          //设置编码格式
?>
```

（2）创建 index.php 文件，完成数据的循环输出。首先，包含数据库连接文件。然后，定义 SQL 查询语句，通过 execute()函数执行查询操作。最后，通过 while 语句循环输出查询结果。

关键代码如下：

```
<?php
     include_once 'conn/conn.php';                                          //载入数据库链接文件
     $sqlstr = 'select * from tb_user where id   limit 5';                  //SQL 查询语句
     $rst = $conn->execute ( $sqlstr ) or die ( 'error：  ' . $conn->errorMsg () );   //执行查询语句
     while ( ! $rst->EOF ) { //wihle 语句循环输出结果
?>
  <tr>
          <td width="126" height="30" align="center">
<?php
          echo $rst->fields ['user'];
?>      </td>
          <td width="117" align="center"><?php
          echo $rst->fields ['dates'];
?>
     </td>
                    </tr>
<?php

          $rst->movenext ();                                                //指针下移
     }
     $rst->close ();                                                        //关闭连接
     $conn->close ();
?>
```

运行上述代码，结果如图 19.8 所示。

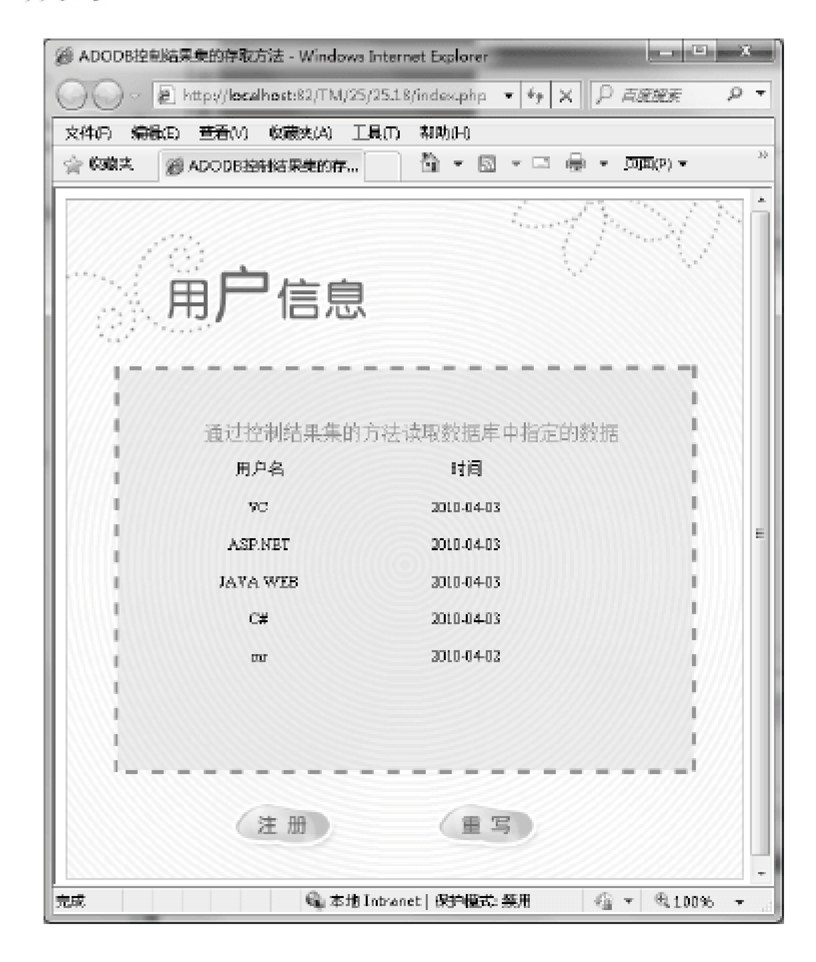


图 19.8　控制结果集存取方法的运行结果

19.6　小　　结

本章介绍了 ADODB 类库的概念、支持的数据库和下载安装的方法，对类库中的常用函数还作了详细的讲解，并且通过实例来帮助读者加深理解。相信读者通过本章的学习，能够熟练地应用 ADODB 类库操作各种数据库。

19.7　学习成果检验

1．使用 ADODB 类库读取 Access 数据库中的数据。**（实例位置：光盘\TM\Instance\19\19.19）**

2．使用 ADODB 类库同时连接多个数据库，并将 Access 数据库表中的数据存储到 MySQL 数据库中。**（实例位置：光盘\TM\Instance\19\19.20）**

第20章

数据库编程技术

（视频讲解：136 分钟）

PHP 所支持的数据库类型较多，在这些数据库中，MySQL 数据库与 PHP 的兼容最好，与 Linux 系统、Apache 服务器和 PHP 语言构成了当今主流的 LAMP 网站架构模式，并且 PHP 提供了多种操作 MySQL 数据库的方式，可以适合不同需求和不同类型的项目。本章将介绍如何通过 MySQL 函数操作数据库，这也是中小型项目常用的方式之一。

通过阅读本章内容，你可以：

- 了解 PHP 访问 MySQL 数据库的步骤
- 掌握常用 MySQL 数据库函数的使用方法
- 掌握 PHP 操作 MySQL 数据库进行增、删、改、查的操作
- 掌握数据分页显示的实现方法
- 掌握应用多种方法获取结果集
- 掌握获取查询结果集中的记录行数

20.1　PHP 访问 MySQL 数据库的一般步骤

和其他语言类似，PHP 操作 MySQL 数据库的过程一般分为 5 步，分别为连接 MySQL 数据库服务器、选择数据库、执行 SQL 语句、关闭结果集以及断开与 MySQL 服务器的连接，如图 20.1 所示。下面将具体介绍其实现过程。

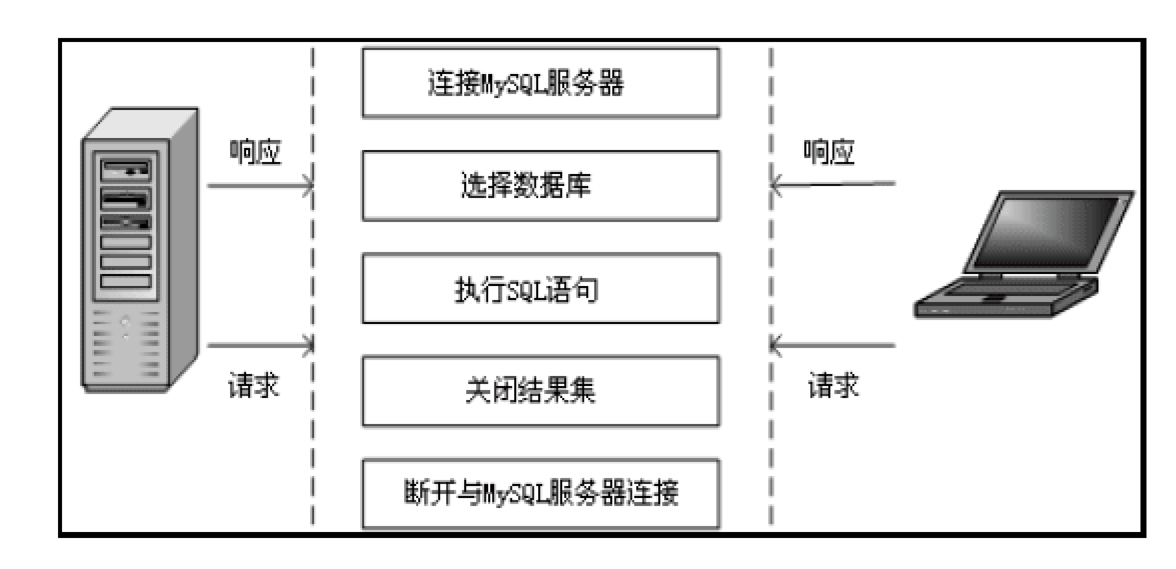


图 20.1　PHP 访问 MySQL 数据库的一般步骤

1．连接 MySQL 服务器

应用 mysql_connect()函数建立与 MySQL 服务器的连接，并返回一个连接标识，在以后对 MySQL 服务器进行操作时，可以根据这个连接标识定位不同的连接。

2．选择数据库

应用 mysql_select_db()函数选择 MySQL 数据库服务器上的数据库，并与该数据库建立连接。

3．执行 SQL 语句

在选择的数据库中应用 mysql_query()函数执行 SQL 语句。对数据的操作主要包括以下 5 种方式。

- ☑ 查询数据：应用 select 语句实现数据的查询功能。
- ☑ 显示数据：应用 select 语句显示数据的查询结果。
- ☑ 插入数据：应用 insert 语句向数据库中插入数据。
- ☑ 更新数据：应用 update 语句修改数据库中的记录。
- ☑ 删除数据：应用 delete 语句删除数据库中的记录。

4．关闭结果集

数据库操作完成后，需要关闭结果集，以释放系统资源。

```
mysql_free_result($result);
```

技巧 如果在多个网页中都要频繁进行数据库访问，则可以建立与数据库服务器的持续连接来提高效率。因为每次与数据库服务器的连接需要较长的时间和较大的资源开销，持续的连接相对来说会更有效。建立持续连接的方法就是在数据库连接时，调用 mysql_pconnect()函数代替 mysql_connect()函数。建立的持续连接在本程序结束时，不需要调用 mysql_close()函数来关闭。下次程序在此执行 mysql_pconnect()函数时，系统自动直接返回已经建立的持续连接的 ID 号，而不再去真的连接数据库。

5．断开与 MySQL 服务器的连接

每使用一次 mysql_connect()或 mysql_query()函数，都会消耗系统资源。这在少量用户访问 Web 网站时影响不明显，但如果用户连接超过一定数量，就会造成系统性能的下降，甚至死机。为了避免这种现象的发生，在完成数据库的操作后，可以应用 mysql_close()函数关闭与 MySQL 服务器的连接，以节省系统资源。

```
mysql_close($link);
```

20.2　PHP 操作 MySQL 数据库的方法

20.1 节简要介绍了 PHP 访问和操作 MySQL 数据库的一般步骤。PHP 提供了大量操作 MySQL 数据库的方式，其中函数方式相对简便并且应用较广泛。本节将详细介绍 PHP 中 MySQL 相关函数的使用方法。

20.2.1　使用 mysql_connect()函数连接 MySQL 服务器

PHP 操作 MySQL 数据库，首先要建立与 MySQL 数据库的连接，PHP 实现与数据库的连接相对简便，只需使用 mysql_connect()函数即可。函数语法如下：

```
resource mysql_connect ( [string server [, string username [, string password [, bool new_link [, int client_flags]]]]] )
```

mysql_connect()函数用于打开一个到 MySQL 服务器的连接，如果成功则返回一个 MySQL 连接标识，失败则返回 false。该函数的参数如表 20.1 所示。

表 20.1　mysql_connect()函数的参数说明

参　数	说　明
server	MySQL 服务器。可以包括端口号，如"hostname:port"；或者到本地套接字的路径，如对于 localhost 的":/path/to/socket"。如果 PHP 指令 mysql.default_host 未定义（默认情况），则默认值是'localhost:3306'
username	用户名。默认值是服务器进程所有者的用户名
password	密码。默认值是空密码
new_link	如果用同样的参数再次调用 mysql_connect()函数，将不会建立新连接，而将返回已经打开的连接标识。参数 new_link 改变此行为并使 mysql_connect()函数总是打开新的连接，即使 mysql_connect()函数曾在前面被用同样的参数调用过
client_flags	client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_SSL，MYSQL_CLIENT_COMPRESS，MYSQL_CLIENT_IGNORE_SPACE 或 MYSQL_CLIENT_INTERACTIVE

例 20.1　应用 mysql_connect()函数创建与 MySQL 服务器的连接，MySQL 数据库服务器地址为 127.0.0.1，用户名为 root，密码为 111。（**实例位置：光盘\TM\Instance\20\20.1**）

代码如下：

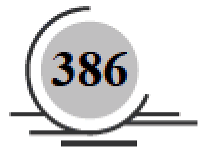
```
<?php
$host = "127.0.0.1";                          //MySQL 服务器地址
$userName = "root";                           //用户名
$password = "111";                            //密码
if ($connID = mysql_connect($host, $userName, $password)){
                                              //建立与 MySQL 数据库的连接，并弹出提示对话框
    echo "<script language='javascript'>alert('数据库连接成功！');</script>";
}else{
    echo "<script language='javascript'>alert('数据库连接失败！');</script>";
}
?>
```

运行上述代码，如果在本地计算机中安装了 MySQL 数据库，并且 root 用户名为 root，密码为 111，则会弹出如图 20.2 所示的对话框。



图 20.2　数据库连接成功

技巧　在项目完成后，为了屏蔽由于数据库连接失败而显示的不友好的错误信息，可以在 mysql_ connect()函数前加“@”，该符号用来屏蔽错误提示。



20.2.2　使用 mysql_select_db()函数选择数据库文件

成功与 MySQL 数据库建立连接后，需要选择 MySQL 数据库服务器中指定的数据库。PHP 中使用 mysql_select_db()函数实现数据库的选择功能。该函数的语法格式如下：

```
bool mysql_select_db ( string database_name [, resource link_identifier] )
```

mysql_select_db()函数用于设定与指定的连接标识符所关联的服务器上的当前激活数据库。如果没有指定连接标识符，则使用上一个打开的连接。如果没有打开的连接，本函数将无参数调用 mysql_connect()函数来尝试打开一个使用。其后的每个 mysql_query()函数调用都会作用于当前激活的数据库。该函数的参数说明如表 20.2 所示。

表 20.2　mysql_select_db()函数的参数说明

参　数	说　明
database_name	必选参数，用户指定要选择的数据库名称
link_identifier	可选参数，数据库连接 ID，如果省略该参数，则默认为最近一次与数据库建立的连接

例 20.2　首先使用 mysql_connect()函数建立与 MySQL 数据库的连接并返回数据库连接 ID，然后使用 mysql_select_db()函数选择 MySQL 数据库服务器中名为 db_database20 的数据库。（**实例位置：光盘\TM\Instance\20\20.2**）

实现代码如下：

```
<?php
$host = "127.0.0.1";                                    //MySQL 服务器地址
$userName = "root";                                     //用户名
$password = "111";                                      //密码
$dbName = "db_database20";                              //数据库
$connID = mysql_connect($host, $userName, $password);  //建立与 MySQL 数据库服务器的连接
if(mysql_select_db($dbName, $connID)){                 //选择数据库
    echo "数据库选择成功！";
}else{
    echo "数据库选择失败！";
}
?>
```

运行上述代码，如果本地 MySQL 数据库服务器中存在名为 db_database20 的数据库，将在页面中显示如图 20.3 所示的提示信息。

20.2.3　使用 mysql_query()函数执行 SQL 语句

成功选择 MySQL 数据库服务器中的数据库后，即可对所选数据库中的数据表进行查询、更改以及删除等操作，PHP 使用 mysql_query()函数就可以实现上述所有操作，操作极其简便，说明在 PHP 底层进行了复杂的封装，而提供给上层开发人员一种简便的编程模式，这也是 PHP 操作简便的体现和应用广泛的原因。mysql_query()函数的语法格式如下：

```
resource mysql_query ( string query [, resource link_identifier] )
```

mysql_query()函数用于执行一条查询语句，该函数的参数说明如表 20.3 所示。

表 20.3 mysql_query()函数的参数说明

参 数	说 明
query	字符串类型，传入的是 SQL 的指令
link_identfier	资源类型，传入的是由 mysql_connect()函数或 mysql1_pconnect()函数返回的连接号。如果省略该参数，则会使用最后一个打开的 MySQL 数据库连接

例 20.3 查询学生信息表中学生的成绩信息。（**实例位置：光盘\TM\Instance\20\20.3**）

代码如下：

```
<?php
$host = "127.0.0.1";                                        //MySQL 数据库服务器
$userName = "root";                                         //用户名
$password = "111";                                          //密码
$dbName = "db_database20";                                  //数据库名
$connID = mysql_connect($host, $userName, $password);       //连接 MySQL 数据库
mysql_select_db($dbName, $connID);                          //选择 MySQL 数据库
mysql_query("set names utf8");                              //设置字符集
echo "<table border=\"1px\" align=\"center\">
          <tr>
               <td>学号</td>
               <td>姓名</td>
               <td>班级</td>
               <td>语文</td>
               <td>数学</td>
               <td>英语</td>
           </tr>";
$query = mysql_query("select sno, sname, class, chinese, math ,english from tb_student", $connID);
                                                            //执行查询
while($result = mysql_fetch_array($query))                  //获取结果集并输出查询结果
{
    echo   "<tr>
              <td>".$result["sno"]."</td>
               <td>".$result["sname"]."</td>
               <td>".$result["class"]."</td>
               <td>".$result["chinese"]."</td>
               <td>".$result["math"]."</td>
               <td>".$result["english"]."</td>
           </tr>";
}
echo "</table>";
?>
```

运行上述实例，结果如图 20.4 所示。

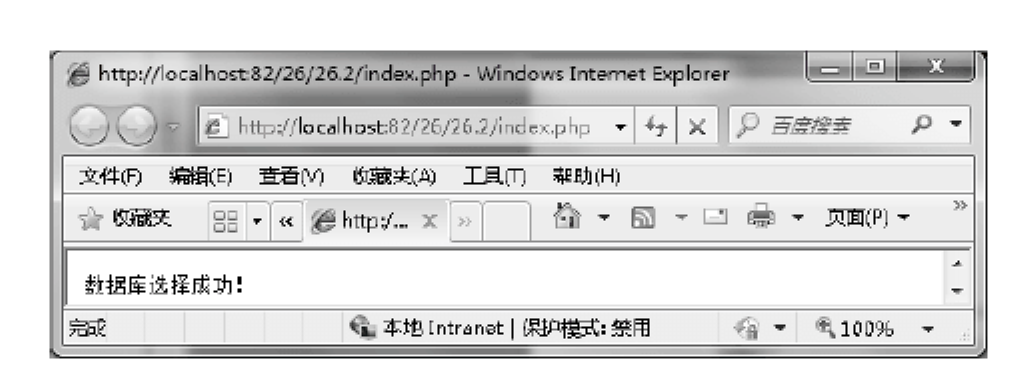


图 20.3 数据库选择成功

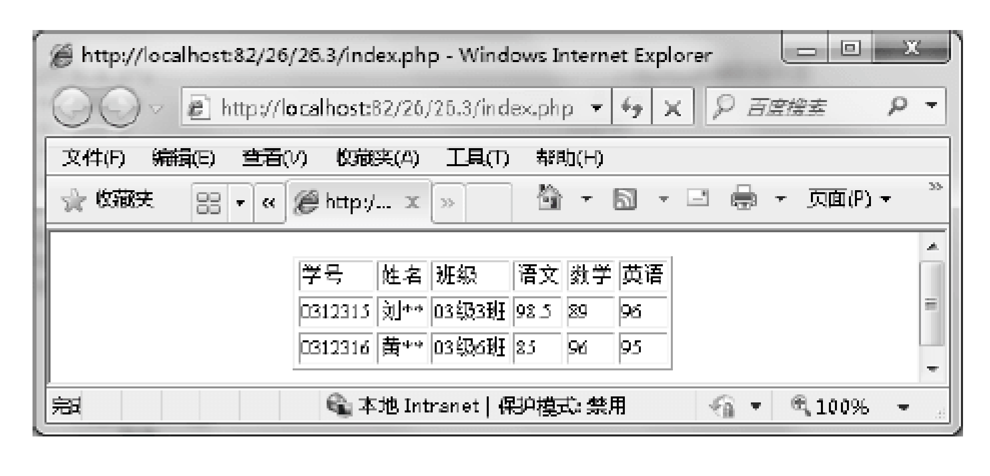


图 20.4 查询学生成绩



20.2.4　应用 mysql_fetch_array()函数从数组结果集中获取信息

视频讲解：光盘\TM\Video\第 20 章\应用 mysql_fetch_array()函数从数组结果集中获取信息.exe

使用 mysql_query()函数执行查询后，并不能返回查询结果，那么如何才能获取查询结果呢？PHP 提供了 mysql_fetch_array()函数实现获取查询结果集的功能。该函数的语法格式如下。

```
array mysql_fetch_array ( resource result [, int result_type] )
```

mysql_fetch_array()函数的参数说明如表 20.4 所示。

表 20.4　mysql_fetch_array()函数的参数说明

参　　数	说　　明
result	资源类型的参数，要传入的是由 mysql_query()函数返回的数据指针
result_type	可选项，整数型参数，要传入的是 MYSQL_ASSOC、MYSQL_NUM、MYSQL_BOTH 3 种由 PHP 定义好的常数之一，默认值是 MYSQL_BOTH 用 MYSQL_ASSOC 只得到关联索引（相当于 mysql_fetch_assoc()函数） 用 MYSQL_NUM 只得到数字索引（相当于 mysql_fetch_row()函数） 用 MYSQL_BOTH 将得到一个同时包含关联和数字索引的数组

例 20.4　按员工编号以模糊查询的方式查询员工信息，并显示全部查询结果。**（实例位置：光盘\TM\Instance\20\20.4）**

具体实现步骤如下所示。

（1）建立与 MySQL 数据库的连接，并返回数据库连接 ID。其代码如下：

```
<?php
$connID=mysql_connect("localhost","root","111");            //建立与数据库的连接
mysql_select_db("db_database20", $connID);                 //选择数据库
mysql_query("set names utf8");                              //设置字符集
?>
```

（2）建立查询信息录入表单，表单及表单元素如表 20.5 所示。

表 20.5　员工信息录入表单

元 素 类 型	元 素 名 称	属 性 设 置	说　　明
表单	form1	name="form1" method="post" action="<?php echo $_SERVER['PHP_SELF']?>"	表单
文本域	number	name="number" type="text" id="number"	录入员工编号
隐藏域	flag	type="hidden" name="flag" value="1"	判断表单是否提交
提交按钮	submit	name="submit" type="submit" value="提交"	提交按钮

（3）使用$_POST 全局数组接收表单提交的 flag 元素的值，并使用 isset()函数判断是否已经设置了该元素的值，如果已设置则说明已经提交了表单，然后采用模糊查询的方式查询所有与查询关键字相匹配的员工信息，并使用 while 循环将查询结果显示出来。其代码如下：

```
<?php
  if(isset($_POST["flag"])){
        $query=mysql_query("select * from tb_employee where number like '%".$_POST["number"]."%'");
        if($query){
              while($myrow=mysql_fetch_array($query)){
?>
  <tr>
```

```
        <td  align="center"  bgcolor="#FFFFFF"  class="STYLE4"><span  class="STYLE2"><?php  echo  $myrow
[number];? > </span></td>
        <td  align="center"  bgcolor="#FFFFFF"  class="STYLE4"><span  class="STYLE2"><?php  echo  $myrow
[name];?>    </span></td>
        <td height="23" align="center" bgcolor="#FFFFFF" class="STYLE4"><span class="STYLE2"><?php echo
$myrow [tel];?></span></td>
        <td height="23" align="center" bgcolor="#FFFFFF" class="STYLE4"><span class="STYLE2"><?php echo
$myrow [address];?></span></td>
    </tr>
  <?php
                }
            }
    }
?>
```

运行该实例，在员工查询信息录入表单中输入员工编号，然后单击“提交”按钮，即可以模糊查询的方式查询出所有与查询关键字相匹配的员工信息，如图 20.5 所示。

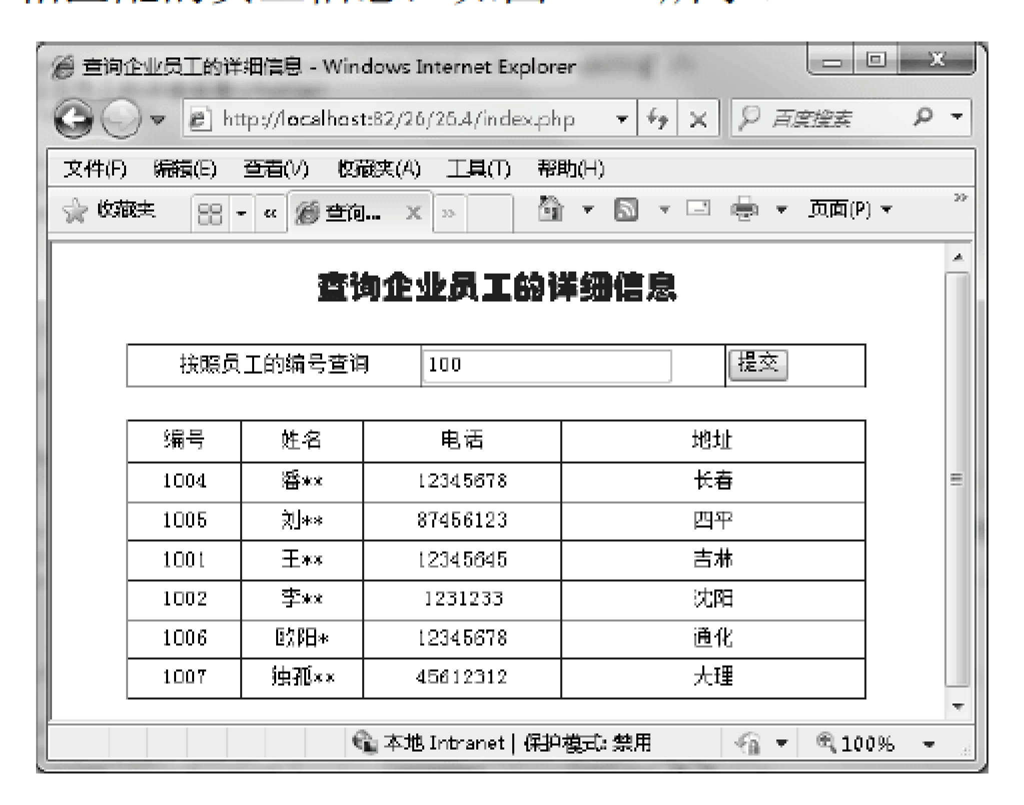


图 20.5　查询员工信息

20.2.5　应用 mysql_fetch_object()函数从结果集中获取一行作为对象

视频讲解：光盘\TM\Video\第 20 章\应用 mysql_fetch_object()函数从结果集中获取一行作为对象.exe

20.2.4 节中讲解了应用 mysql_fetch_array()函数来获取结果集中的数据。除了这个方法以外，应用 mysql_fetch_object()函数也可以轻松实现这一功能，下面通过同一个实例的不同方法来体验一下这两个函数在使用上的区别。首先介绍 mysql_fetch_object()函数。

语法如下：

```
object   mysql_fetch_object ( resource result )
```

mysql_fetch_object()函数和 mysql_fetch_array()函数类似，只有一点区别：前者返回一个对象而不是数组，即该函数只能通过字段名来访问数组。访问结果集中行元素的语法结构如下：

```
$row->col_name                                    //col_name 为列名，$row 代表结果集
```

例如，如果从某数据表中检索 id 和 name 值，可以用$row->id 和$row-> name 访问行中的元素值。



本函数返回的字段名是区分大小写的，这是初学者学习时最容易忽视的问题。

例 20.5　使用 mysql_fetch_object()函数获取查询到的图书信息。（**实例位置：光盘\TM\Instance\20\20.5**）

具体开发步骤如下。



（1）建立图书查询表单，表单及表单元素说明如表 20.6 所示。

表 20.6　图书信息查询表单及表单说明

元素类型	元素名称	属性设置	说　明
表单	myform	name="myform" method="post" action=""	表单
文本域	txt_book	name="txt_book" type="text" id="txt_book" size="25"	查询关键字
提交按钮	submit	type="submit" name="Submit" value="查询"	查询按钮

（2）建立与 MySQL 数据库的连接，设置字符集，并返回数据库连接 ID。其代码如下：

```
$link=mysql_connect("localhost","root","111") or die("数据库连接失败".mysql_error());    //建立与数据库的连接
mysql_select_db("db_database20",$link);                                                  //选择数据库
mysql_query("set names gb2312");                                                         //设置字符集
```

（3）应用 mysql_query()函数执行 SQL 查询语句，并使用 mysql_fetch_object()函数获取查询语句的结果集。代码如下：

```
if ($_POST[Submit]=="查询"){
    $txt_book=$_POST[txt_book];                                              //接收查询关键字
    $sql=mysql_query("select * from tb_book where bookname like '%".trim($txt_book)."%'");
          //如果选择的条件为“like”，则进行模糊查询
     $info=mysql_fetch_object($sql);
}
```

（4）应用 do…while 循环语句以对象的方式输出结果集中的图书信息到浏览器中。其代码如下：

```
do{
?>
<tr align="left" bgcolor="#FFFFFF">
  <td height="20" align="center"><?php echo $info->id; ?></td>
  <td > <?php echo $info->bookname; ?></td>
  <td align="center"><?php echo $info->issuDate; ?></td>
  <td align="center"><?php echo $info->price; ?></td>
  <td align="center"> <?php echo $info->maker; ?></td>
  <td> <?php echo $info->publisher; ?></td>
</tr>
<?php
}while($info=mysql_fetch_object($sql));
```

保存 index.php 动态页，在 IE 浏览器中输入地址，按 Enter 键，运行结果如图 20.6 所示。

20.2.6　应用 mysql_fetch_row()函数逐行获取结果集中的每条记录

视频讲解：光盘\TM\Video\第 20 章\应用 mysql_fetch_row()函数逐行获取结果集中的每条记录.exe

前面介绍了应用 mysql_fetch_array()和 mysql_fetch_object()函数来获取结果集中的数据。本节向读者介绍第 3 种方法，应用 mysql_fetch_row()函数逐行获取结果集中的每条记录。首先来了解 mysql_ fetch_row()函数。

语法如下：

```
array mysql_fetch_row ( resource result )
```

mysql_query()函数将查询语句发送到服务器中执行。查询语句不使用分号终止。如果查询非法或由于某些原因不能执行，则 mysql_query()函数返回 false，否则返回一个结果集标识符。mysql_fetch_row()函数从和指定的结果标识关联的结果集中获取一行数据并作为数组返回，将此行赋给变量$row，每个结果的列储存在一个数组的单元中，偏移量从 0 开始，即以$row[0]的形式访问第一个元素（只有一个元素时也是如此）。依次调用 mysql_fetch_row()函数将返回结果集中的下一行，直到没有更多行则返回 false。

例 20.6　查询图书信息，并使用 mysql_fetch_row()函数获取结果集显示图书信息。（**实例位置：光盘\TM\Instance\20\20.6**）

具体开发步骤如下。

（1）创建项目、添加表单、连接 MySQL 服务器以及设置默认数据库的实现过程与例 20.5 开发步骤中的（1）～（2）相同，这里就不再赘述了。

（2）在应用 mysql_query()函数执行 SQL 查询语句后，与实例 20.5 不同的是，本实例使用 mysql_fetch_row()函数获取查询语句的结果集。其代码如下：

```
<?php
$sql=mysql_query("select * from tb_book");
$row=mysql_fetch_row ($sql);
if ($_POST[Submit]=="查询"){
$txt_book=$_POST[txt_book];
//如果选择的条件为“like”，则进行模糊查询
$sql=mysql_query("select * from tb_book where bookname like '%". trim($txt_book)."%'");
$row=mysql_fetch_row($sql);
}
?>
```

（3）应用 if 条件语句对结果集变量$info 进行判断，如果该值为假，则检索的图书信息不存在，应用 echo 语句输出提示信息。其代码如下：

```
<?php
if($row==false){                        //如果检索的信息不存在，则输出相应的提示信息
    echo "<div align='center' style='color:#FF0000; font-size:12px'>对不起，您检索的图书信息不存在!</div>";
}
?>
```

（4）应用 do…while 循环语句以对象的方式输出结果集中的图书信息到浏览器中。其代码如下：

```
<?php
do{
?>
<tr align="left" bgcolor="#FFFFFF">
  <td height="20" align="center"><?php echo $row[0]; ?></td>
  <td > <?php echo $row[1]; ?></td>
  <td align="center"><?php echo $row[2]; ?></td>
  <td align="center"><?php echo $row[3]; ?></td>
  <td align="center"> <?php echo $row[4]; ?></td>
  <td> <?php echo $row[5]; ?></td>
</tr>
<?php
}while($row=mysql_fetch_row($sql));
?>
```

保存 index.php 动态页，在 IE 浏览器中输入地址，按 Enter 键，运行结果如图 20.7 所示。

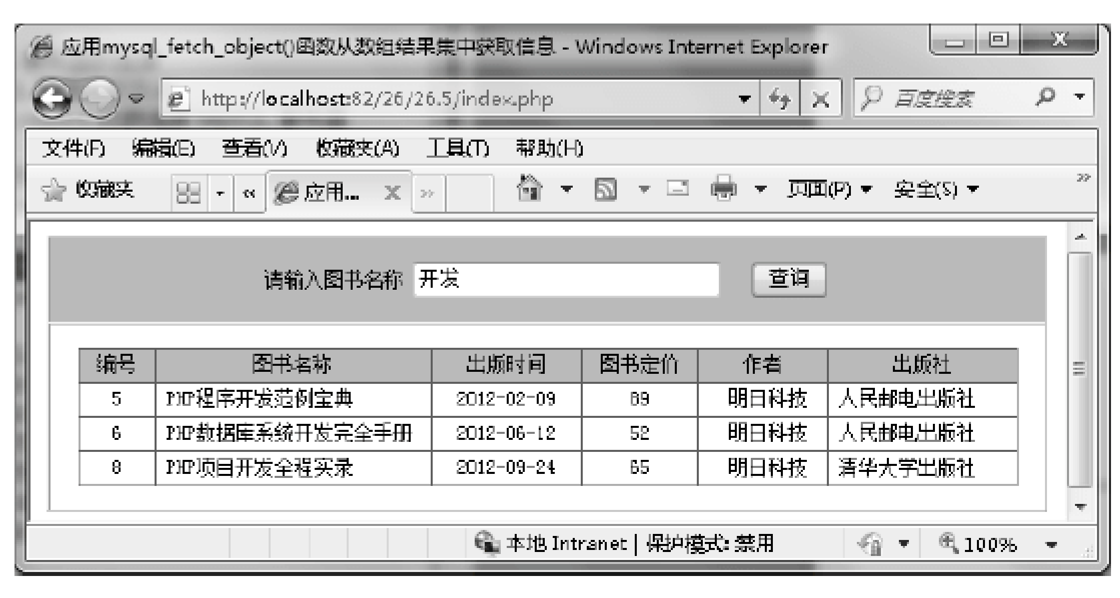


图 20.6　查询图书信息

图 20.7　使用 mysql_fetch_row()函数获取结果集查询图书信息



20.2.7　应用 mysql_num_rows()函数获取查询结果集中的记录数

视频讲解：光盘\TM\Video\第 20 章\应用 mysql_num_rows()函数获取查询结果集中的记录数.exe

要获取由 select 语句查询到的结果集中行的数目，必须使用 mysql_num_rows()函数实现。首先来看一下该函数的语法结构。

语法如下：

```
int mysql_num_rows ( resource result )
```

注意 使用 mysql_unbuffered_query()函数查询到的数据结果，无法使用 mysql_num_rows()函数来获取查询结果的记录数。

例 20.7　使用 mysql_num_rows()函数获取查询结果的记录数。（**实例位置：光盘\TM\Instance\20\20.7**）

具体开发步骤如下。

（1）建立与 MySQL 数据库的连接，设置字符集为 GB2312，并返回数据库连接 ID。其代码如下：

```
$link=mysql_connect("localhost","root","111") or die("数据库连接失败".mysql_error());    //建立连接
mysql_select_db("db_database20",$link);                                              //选择数据库
mysql_query("set names gb2312");                                                     //设置字符集
```

（2）默认情况下显示所有图书信息。其代码如下：

```
$sql=mysql_query("select * from tb_book");                                           //查询所有图书信息
$info=mysql_fetch_object($sql);                                                      //获取结果集
```

（3）如果用户输入了查询关键字，并单击了“查询”按钮，则采用模糊查询的方式显示出所有符合条件的记录。其代码如下：

```
if ($_POST[Submit]=="查询"){
    $txt_book=$_POST[txt_book];
    $sql=mysql_query("select * from tb_book where bookname like '%".trim($txt_book)."%'");
    //如果选择的条件为“like”，则进行模糊查询
    $info=mysql_fetch_object($sql);
}
if($info==false){                                        //如果检索的信息不存在，则输出相应的提示信息
     echo "<div align='center' style='color:#FF0000; font-size:12px'>对不起，您检索的图书信息不存在!</div>";
}
do{
?>
<tr align="left" bgcolor="#FFFFFF">
    <td height="20" align="center"><?php echo $info->id; ?></td>
    <td > <?php echo $info->bookname; ?></td>
    <td align="center"><?php echo $info->issuDate; ?></td>
    <td align="center"><?php echo $info->price; ?></td>
    <td align="center"> <?php echo $info->maker; ?></td>
    <td> <?php echo $info->publisher; ?></td>
</tr>
<?php
    }while($info=mysql_fetch_object($sql));
?>
```

在 IE 浏览器中输入地址，按 Enter 键，程序默认输出图书信息表中的全部图书信息，并自动汇总记录条数，如图 20.8 所示。在文本框中输入要检索的图书名称，如“开发”（支持模糊查询，程序自动去除查询关键字的左右空格），单击“查询”按钮，即可按条件检索指定的图书信息到浏览器，并自动汇总检索

到的记录条数，运行结果如图 20.9 所示。

在默认页应用 mysql_num_rows()函数获取数据表的总记录数

图 20.8　默认统计数据表中所有记录

应用 mysql_num_rows()函数获取查询结果集中的总记录数

图 20.9　应用 mysql_num_rows()函数获取查询结果集中的记录数

注意 如果要获取由 insert、update、delete 语句所影响到的数据行数，必须应用 mysql_affected_ rows()函数来实现。

20.2.8　关闭连接

视频讲解：光盘\TM\Video\第 20 章\关闭连接.exe

完成对数据库的操作后，需要及时断开与数据库的连接并释放内存，否则会浪费大量的内存空间，在访问量较大的 Web 项目中，很可能导致服务器崩溃。在 MySQL 函数库中，使用 mysql_close()函数断开与 MySQL 服务器的连接。该函数的语法格式如下：

```
bool mysql_close ( [resource link_identifier] )
```

mysql_close()函数用于关闭指定的连接标识所关联的 MySQL 服务器的非持久连接。如果没有指定 link_identifier 参数，则关闭最后一个打开的连接。

例 20.8　使用 mysql_connect()函数建立与数据库的连接，然后使用 mysql_select_db()函数选择数据库并使用 mysql_close()函数断开与 MySQL 数据库的连接，在断开与 MySQL 数据库连接后，再次使用 mysql_ select_db()函数选择数据库，从两次选择数据库的情况来判断 mysql_close()函数是否能起到断开数据库连接的作用。**（实例位置：光盘\TM\Instance\20\20.8）**

实例代码如下：

```
<?php
$host = "127.0.0.1";                                    //MySQL 数据库服务器
$userName = "root";                                     //用户名
$password = "111";                                      //密码
$dbName = "db_database20";                              //数据库名
$connID = mysql_connect($host, $userName, $password);   //连接 MySQL 数据库
mysql_select_db($dbName, $connID);                      //选择数据库
mysql_close($connID);                                   //断开与数据库连接
mysql_select_db($dbName, $connID);                      //再次选择数据库
?>
```

运行本实例，将在页面中输出如图 20.10 所示的错误信息。从错误信息中可以判断，第一次对数据库选择操作是成功的，而第二次操作是失败的，即可证明 mysql_close()函数成功关闭了与数据库的连接。

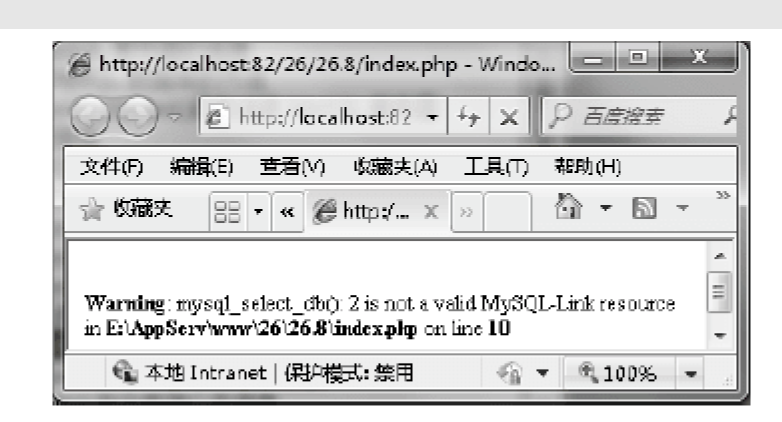


图 20.10　错误提示

20.3 管理 MySQL 数据库中的数据

PHP 操作 MySQL 数据库是 PHP 中基础且较重要的技术，开发人员只有熟练地掌握这部分知识才能够独立开发出基于 PHP 的数据库应用。

本节通过一个公告栏进一步讲解 PHP 通过使用 MySQL 操作函数实现对 MySQL 数据库进行增、删、改、查操作的具体实现过程。

20.3.1 应用 insert 命令动态添加公告信息

视频讲解：光盘\TM\Video\第 20 章\应用 insert 命令动态添加公告信息.exe

在实现动态添加公告信息前，首先要建立数据库及数据表结构。创建数据表的方法有两种，一种是在命令提示符下创建，另一种是使用 phpMyAdmin 图形化管理工具创建。下面介绍这两种创建数据库的具体步骤和实现过程。

☑ 在命令提示符下创建数据表结构

选择“开始”/“运行”命令，打开“运行”窗口。在“运行”窗口中输入“cmd”命令将弹出命令提示符窗口。在命令提示符下使用如下命令登录 MySQL 数据库服务器：

```
mysql –uroot –p111
```

成功登录 MySQL 数据库服务器后，使用 use 命令选择数据库 db_database20，命令如下：

```
use db_database20
```

选择数据库 db_database20 后，输入如下命令创建数据表 tb_affiche：

```
CREATE TABLE 'tb_affiche' (
    'id' INT( 4 ) NOT NULL AUTO_INCREMENT PRIMARY KEY ,
    'title' VARCHAR( 200 ) CHARACTER SET gb2312 COLLATE gb2312_chinese_ci NOT NULL ,
    'content' TEXT CHARACTER SET gb2312 COLLATE gb2312_chinese_ci NOT NULL ,
    'createtime' DATETIME NOT NULL
) ENGINE = MYISAM ;
```

☑ 利用 phpMyAdmin 图形化管理工具创建数据表结构

在 phpMyAdmin 图形化管理工具中，选择数据库 db_database20，然后按图 20.11 所示的表结构创建 tb_affiche 表。

服务器: localhost ▸ 数据库: db_database20 ▸ 表 : tb_affiche

浏览 结构 SQL 搜索 插入 导出 Import 操作 清空 删除

字段	类型	整理	属性	Null	默认	额外	操作
id	int(4)			否		auto_increment	
title	varchar(200)	gb2312_chinese_ci		否			
content	mediumtext	gb2312_chinese_ci		否			
createtime	datetime			否			

图 20.11 公告信息数据表 tb_affiche 的字段说明

下面按照 PHP 访问 MySQL 数据库的一般步骤讲解使用 PHP 中的 MySQL 操作函数向数据库表添加数据的方法。

例 20.9 应用 insert 命令动态地向数据库中添加公告信息，然后应用 mysql_query()函数将 SQL 语句发送到 MySQL 服务器，从而完成将数据动态添加到数据库的操作。（**实例位置：光盘\TM\Instance\20\20.9**）

程序开发步骤如下。

（1）在左侧导航区的添加公告信息的图片上添加热区。其代码如下：

```
<img src="images/image_09.gif" width="202" height="310" border="0" usemap="#Map">
<map name="Map">
  <area shape="rect" coords="30,45,112,63" href=" add_affiche.php">
</map>
```

（2）创建 add_affiche.php 页面，在该页面中建立公告信息录入表单，并指定表单的提交地址为 check_add_affiche.php 页面。其代码如下：

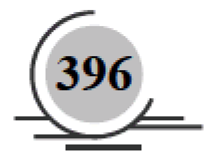
```
<form name="form1" method="post" action="check_add_affiche.php">
   <table width="520" height="212"   border="0" cellpadding="0" cellspacing="0" bgcolor="#FFFFFF">
     <tr>
        <td width="87" align="center">公告主题：</td>
        <td width="433" height="31">
        <input name="txt_titile" type="text" id="txt_titile" size="40">
         * </td>
      </tr>
      <tr>
        <td height="124" align="center">公告内容：</td>
        <td><textarea name="txt_content" cols="50" rows="8" id="txt_content"></textarea></td>
      </tr>
      <tr>
        <td height="40" colspan="2" align="center">
        <input  name="Submit"  type="submit"  class="btn_grey"  value=" 保 存 "  onClick="return
check(form1)">
        <input type="reset" name="Submit2" value="重置">
        </td>
      </tr>
   </table>
</form>
```

提交表单前，要对用户输入数据的合法性进行验证，以便提高系统的安全性。公告录入表单中所要录入的数据只需验证是否为空即可。该部分是采用 JavaScript 脚本在客户端进行验证的，这可以有效降低服务器的负载。实现代码如下：

```
<script language="javascript">
function check(form){                                    //验证表单信息是否为空
     if(form.txt_title.value==""){                       //公告标题信息不能为空，否则弹出提示信息
        alert("请输入公告标题!");form.txt_title.focus();return false;
     }
     if(form.txt_content.value==""){                     //公告内容不能为空，否则弹出提示信息
        alert("请输入公告内容!");form.txt_content.focus();return false;
     }
form.submit();                                           //提交表单信息到数据处理页
}
</script>
```

（3）提交表单信息到数据处理页 check_add_affiche.php，连接 MySQL 数据库服务器，并指定数据库的编码格式为 GB2312。通过 POST 方法获取表单传递过来的值，然后应用 insert 命令将表单信息添加到数据表，并由 mysql_query()函数发送到服务器，从而完成公告信息的添加。添加成功则弹出提示信息，并重新定位到首页 index.php。其代码如下：

```
<?php
  $conn=mysql_connect("localhost","root","111") or die("数据库服务器连接错误".mysql_error());
  mysql_select_db("db_database20",$conn) or die("数据库访问错误".mysql_error());
  /**************这里必须指定数据库的编码方式为 GB2312，否则存储到数据表中的公告信息为乱码*************/
  mysql_query("set names gb2312");                       //选择编码格式为 GB2312
  $title=$_POST[txt_title];                              //获取公告标题信息
```



```
  $content=$_POST[txt_content];                              //获取公告内容
  $createtime=date("Y-m-d H:i:s");                           //获取系统的当前时间，“H”表示 24 小时制必须大写
/************************应用 mysql_query()函数执行 insert…into 语句发送到服务器**********************/
  $sql=mysql_query("insert into tb_affiche(title,content,createtime)values('$title','$content','$createtime')");
  echo "<script>alert('公告信息添加成功!');window.location.href= add_affiche.php';</script>";
  mysql_free_result($sql);                                   //关闭记录集
  mysql_close($conn);                                        //关闭 MySQL 数据库服务器
?>
```

在上面的代码中，date()函数用来获取系统的当前时间，在公告信息添加成功后，应用 JavaScript 脚本弹出提示对话框，并在 JavaScript 脚本中应用“window.location.href='index.php”重定位网页，这里将网页重新定位到首页。

（4）在 IE 浏览器中输入地址，按 Enter 键，单击“添加公告信息”超链接，在页面中添加公告主题和公告内容，单击“保存”按钮，弹出“公告信息添加成功”提示信息，运行结果如图 20.12 所示。单击“确定”按钮，重新定位到首页。

20.3.2 应用 select 命令查询公告信息

视频讲解：光盘\TM\Video\第 20 章\应用 select 命令查询公告信息.exe

实现添加公告信息后，即可对公告信息执行查询操作。

例 20.10 应用 select 命令动态检索数据库中的公告信息，使用 mysql_query()函数将 SQL 语句发送到 MySQL 服务器，从而完成数据的检索操作。（**实例位置：光盘\TM\Instance\20\20.10**）

程序开发步骤如下。

（1）建立导航页面 menu.php，然后应用 include 语句将菜单导航页面 menu.php 嵌入 index.php 页面中。

（2）建立 search_affiche.php 页面，在 search_affiche.php 页的右侧信息显示区域添加一个表单、一个文本框、一个提交按钮。其代码如下：

```
<form name="form1" method="post" action="">
    查询关键字 
    <input name="txt_keyword" type="text" id="txt_keyword" size="40">
    <input type="submit" name="Submit" value="搜索" onClick="return check(form)">
</form>
```

为防止用户搜索空信息，本程序在保存按钮的 onclick 事件中，调用一个由 JavaScript 脚本定义的 check()函数，用来限制文本框信息不能为空。当用户单击“保存”按钮时，自动调用 check()函数。该函数的代码如下：

```
<script language="javascript">
function check(form){                                       //验证表单信息是否为空
     if(form.txt_keyword.value==""){                        //查询关键字不能为空，否则弹出提示信息
          alert("请输入查询关键字!");form.txt_keyword.focus();return false;
     }
form.submit();                                              //提交表单信息
}
</script>
```

（3）提交表单信息到数据处理页 check_add_affiche.php，连接 MySQL 数据库服务器，并指定数据库的编码格式为 GB2312。通过 POST 方法获取表单传递过来的值，然后应用 insert 命令将表单信息添加到数据表，从而完成公告信息的添加。添加成功则弹出提示信息，并重新定位到首页 index.php。其代码如下：

```
<?php
  $conn=mysql_connect("localhost","root","111") or die("数据库服务器连接错误".mysql_error());
  mysql_select_db("db_database20",$conn) or die("数据库访问错误".mysql_error());
  mysql_query("set names gb2312");                          //选择编码格式为 GB2312
```

```
$keyword=$_POST[txt_keyword];                              //获取查询关键字内容
$sql=mysql_query("select * from tb_affiche where title like '%$keyword%' or content like '%$keyword%'");
$row=mysql_fetch_object($sql);                             //获取查询结果集
if(!$row){                                                 //如果未检索到信息资源，则弹出提示信息
echo "<font color='red'>您搜索的信息不存在，请使用类似的关键字进行检索!</font>";
}
do{                                                        //应用 do…while 循环语句输出查询结果
?>
 <tr bgcolor="#FFFFFF">
   <td><?php echo $row->title;?></td>
   <td><?php echo $row->content;?></td>
</tr>
<?php
}while($row=mysql_fetch_object($sql));                     //do…while 循环语句结束
mysql_free_result($sql);                                   //关闭记录集
mysql_close($conn);                                        //关闭 MySQL 数据库服务器
?>
```

（4）在 IE 浏览器中输入地址，按 Enter 键，单击“查询公告信息”超链接，将弹出如图 20.13 所示的查询公告信息页面，在页面的文本框中输入查询关键字，然后单击“搜索”按钮即可查询到所有与查询关键字匹配的公告信息。



图 20.12　添加公告页面的运行结果



图 20.13　查询公告页面的运行结果

20.3.3　解决截取公告主题乱码问题

视频讲解：光盘\TM\Video\第 20 章\解决截取公告主题乱码问题.exe

在开发 Web 程序时，为了保持整个页面的合理布局，经常需要对一些较长的主题进行截取输出，但由于汉字占有两个字符，如果截取位置选取不当，将会导致截取的字符串末尾出现乱码。例如，应用 substr()函数对普通字符进行截取操作会非常方便，但对全角字符进行截取可能会出现乱码。可以通过自定义函数 chinesesubstr()解决上述问题。解决截取字符串乱码前后的对比如图 20.14 和图 20.15 所示。

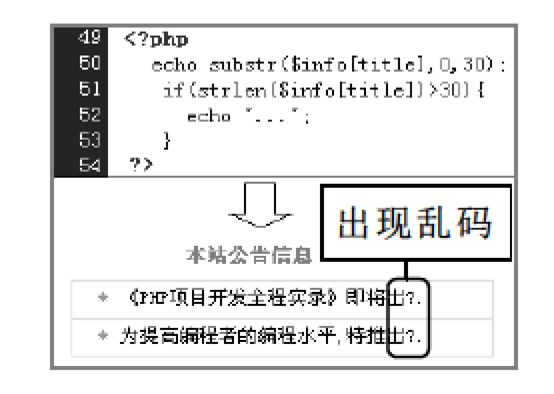


图 20.14　应用 substr()函数截取字符串出现乱码

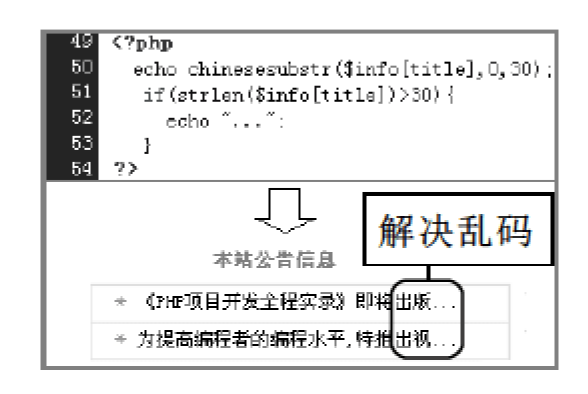
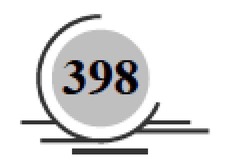


图 20.15　应用自定义函数 chinesesubstr()解决截取字符串乱码



例 20.11　应用自定义函数 chinesesubstr()解决截取公告主题出现乱码的问题。**（实例位置：光盘\TM\Instance\20\20.11）**

程序开发步骤如下。

（1）在菜单导航页 menu.php 中插入一张图片 image_09.gif，在图片上添加热区，链接文件为 intercept.php。然后应用 include 语句将菜单导航页 menu.php 嵌入 index.php 页面中。

（2）建立 intercept.php 页，在 intercept.php 页的右侧信息显示区域动态输出公告主题信息，并截取公告主题的前 30 个字符。其代码如下：

```
<?php
    $conn=mysql_connect("localhost","root","111") or die("数据库服务器连接错误".mysql_error());
    mysql_select_db("db_database20",$conn) or die("数据库访问错误".mysql_error());
    mysql_query("set names gb2312");                  //选择编码格式为 GB2312
    $sql=mysql_query("select * from tb_affiche order by createtime desc limit 0,6");
    $info=mysql_fetch_array($sql);                    //获取查询结果集
    if($info==false){                                 //如果未检索到信息资源，则弹出提示信息
        echo "本站暂无公告信息!";
    }else{
        do{
 ?>
   <tr bgcolor="#E3E3E3">
     <td height="24" align="left" bgcolor="#FFFFFF"><img src="images/xing.gif" width="9" height="9">
     <?php
     //调用自定义函数 chinesesubstr()，屏蔽乱码
     echo chinesesubstr($info[title],0,30);
     if(strlen($info[title])>30){                     //如果公告主题的长度大于 30 个字符，则输出"…"
       echo "...";
     }
     ?>
     </td>
     </tr>
    <?php
        }while($info=mysql_fetch_array($sql));        //do…while 循环语句结束
    }
    mysql_free_result($sql);                          //关闭记录集
    mysql_close($conn);                               //关闭 MySQL 数据库服务器
?>
```

在上面的代码中，应用了自定义函数 chinesesubstr()，为了管理和维护方便，这里将该函数封装在一个独立的 function.php 页中。因此，调用时需要应用 include 语句进行引用。其代码如下：

```
<?php   include("function.php"); ?>
```

其中，function.php 页中定义一个用于截取一段字符串的 chinesesubstr()函数。其代码如下：

```
<?php
 function chinesesubstr($str,$start,$len) { //$str 指字符串，$start 指字符串的起始位置，$len 指字符串长度
    $strlen=$start+$len;                //用$strlen 存储字符串的总长度，即从字符串的起始位置到末尾的总长度
    for($i=0;$i<$strlen;$i++) {
        if(ord(substr($str,$i,1))>0xa0) {   //如果字符串中首个字节的 ASCII 序数值大于 0xa0，则表示汉字
            $tmpstr.=substr($str,$i,2);     //每次取出两位字符赋给变量$tmpstr，即等于一个汉字
            $i++;                           //变量自加 1
        }
          else
            $tmpstr.=substr($str,$i,1);     //如果不是汉字，则每次取出一位字符赋给变量$tmpstr
    }
    return $tmpstr;                         //返回字符串
```

```
}
?>
```

（3）在 IE 浏览器中输入地址，按 Enter 键，系统自动按添加公告的时间降序排列，应用 limit 语句取出最新的 6 条公告主题信息，每条公告主题仅截取前 30 个字符进行输出，公告主题大于 30 个字符的字符串部分用省略号（…）代替，运行结果如图 20.16 所示。如果未检索到相匹配的公告信息，则输出提示信息。



图 20.16　解决截取公告主题乱码问题

20.3.4　分页显示公告信息

视频讲解：光盘\TM\Video\第 20 章\分页显示公告信息.exe

在实现了添加公告信息后，便可以对公告信息执行查询操作。下面在实例 20.9 的基础上实现查询公告信息。

例 20.12　应用 select 命令动态检索数据库中的公告信息，并应用 mysql_query()函数将 SQL 语句发送到 MySQL 服务器，从而完成数据的检索操作。**（实例位置：光盘\TM\Instance\20\20.12）**

程序开发步骤如下。

（1）将菜单导航部分封装成一个独立的页 menu.php，在该页中插入一张图片 image_09.gif，在图片上添加热区，链接 page_affiche.php 文件。

（2）在 page_affiche.php 页的右侧信息显示区域，连接 MySQL 数据库服务器，连接数据库，并指定数据库的编码格式为 GB2312 编码。应用 select 命令将检索公告信息表中的数据，并由 mysql_query()函数发送到服务器，从而完成公告信息的分页显示。其代码如下：

```
<table width="550" border="1" cellpadding="1" cellspacing="1" bordercolor="#FFFFFF" bgcolor="#999999">
 <tr align="center" bgcolor="#f0f0f0">
    <td width="221">公告标题</td>
    <td width="329">公告内容</td>
</tr>
<?php
  $conn=mysql_connect("localhost","root","111") or die("数据库服务器连接错误".mysql_error());
  mysql_select_db("db_database20",$conn) or die("数据库访问错误".mysql_error());
  /**************这里必须指定数据库的编码方式为 GB2312，否则输出到浏览器中的公告信息为乱码***** ********/
  mysql_query("set names gb2312");
  /**************************$page 为当前页，如果$page 为空，则初始化为 1***************************/
  if ($page==""){
     $page=1;}
     if (is_numeric($page)){                              //判断变量$page 是否为数字，如果是则返回 true
     $page_size=4;                                        //每页显示 4 条记录
     $query="select count(*) as total from tb_affiche   order by id desc";
     $result=mysql_query($query);                         //查询符合条件的记录总条数
```



```
    $message_count=mysql_result($result,0,"total");          //要显示的总记录数
    /**********************由记录总数除以每页显示的记录数求出所分的页数***********************/
    $page_count=ceil($message_count/$page_size);
    $offset=($page-1)*$page_size;                            //计算下一页从第几条数据开始循环
    $sql=mysql_query("select * from tb_affiche order by id desc limit $offset, $page_size");
    $row=mysql_fetch_object($sql);                           //获取查询结果集
    if(!$row){                                               //如果未检索到信息资源，则输出提示信息
        echo "<font color='red'>暂无公告信息!</font>";
    }
    do{
    ?>
     <tr bgcolor="#FFFFFF">
       <td><?php echo $row->title;?></td>
       <td><?php echo $row->content;?></td>
     </tr>
<?php
    }while($row=mysql_fetch_object($sql));
}
?>
</table>
```

注意 do…while()循环和 while()循环的区别：do…while()循环是先执行{}中的代码段，然后判断 while 中的条件表达式是否成立，如果返回 true，则重复输出{}中的内容，否则结束循环，执行 while 下面的语句；while 循环是先判断 while 中的表达式，当返回 true 时，再执行“{}”中的代码。两者的主要区别在于，do…while()循环比 while()循环多输出一次结果。

（3）添加一个一行一列的表格，添加如下代码，实现翻页功能。

```
<table width="550" border="0" cellspacing="0" cellpadding="0">
  <tr>
  <!--  翻页条  -->
    <td width="37%">  页次：<?php echo $page;?>/<?php echo $page_count;?>页 记
录：<?php echo $message_count;?> 条  </td>
    <td width="63%" align="right">
    <?php
    /*  如果当前页不是首页  */
    if($page!=1){
    /*  显示“首页”超链接  */
    echo  "<a href=page_affiche.php?page=1>首页</a> ";
    /*  显示“上一页”超链接  */
    echo "<a href=page_affiche.php?page=".($page-1).">上一页</a> ";
    }
    /*  如果当前页不是尾页  */
    if($page<$page_count){
    /*  显示“下一页”超链接  */
    echo "<a href=page_affiche.php?page=".($page+1).">下一页</a> ";
    /*  显示“尾页”超链接  */
    echo  "<a href=page_affiche.php?page=".$page_count.">尾页</a>";
    }
    mysql_free_result($sql);                        //关闭记录集
    mysql_close($conn);                             //关闭 MySQL 数据库服务器
    ?>
```

```
    </tr>
  </table>
```

（4）在 IE 浏览器中输入地址，按 Enter 键，分页显示公告信息，运行结果如图 20.17 所示。

20.3.5 应用 update 命令动态编辑公告信息

视频讲解：光盘\TM\Video\第 20 章\应用 update 命令动态编辑公告信息.exe

公告信息也不是一成不变的，可以对公告主题及公告内容的动态信息进行编辑。

例 20.13 应用 update 命令动态编辑数据库中的公告信息，然后通过 mysql_query()函数将 SQL 语句发送到 MySQL 服务器，从而完成数据的编辑操作。**（实例位置：光盘\TM\Instance\20\20.13）**

程序开发步骤如下。

（1）在菜单导航页 menu.php 的 image_09.gif 图片上添加热区，链接 update_affiche.php 文件。

> **注意** 为了提高效率，将首页 index.php 另存为 update_affiche.php。

（2）在 update_affiche.php 页的右侧信息显示区域应用 select 命令检索出全部的公告信息，添加链接文件 modify.php，并向 modify.php 页传递公告 ID 的参数值。其代码如下：

```
<a href="modify.php?id=<?php echo $row->id;?>">
     <img src="images/update.gif" width="20" height="18" border="0">
</a>
```

（3）按照首页的风格创建一个编辑公告信息的 modify.php 页，在该页面中添加一个表单、一个文本框、一个编辑框、一个隐藏域、一个提交按钮和一个重置按钮，设置表单的 action 属性值为 check_modify_ok.php。其代码如下：

```
<?php
$conn=mysql_connect("localhost","root","111") or die("数据库服务器连接错误".mysql_error());
mysql_select_db("db_database20",$conn) or die("数据库访问错误".mysql_error());
mysql_query("set names gb2312");
$id=$_GET[id];                                                   //应用 GET 方法接收欲编辑的公告 ID
$sql=mysql_query("select * from tb_affiche where id=$id");      //检索公告 ID 所对应的公告信息
$row=mysql_fetch_object($sql);                                  //获取结果集
?>
<form name="form1" method="post" action="check_modify_ok.php">
  <table width="520" height="212"   border="0" cellpadding="0" cellspacing="0" bgcolor="#FFFFFF">
     <tr>
        <td width="87" align="center">公告主题：</td>
        <td width="433" height="31">
        <input name="txt_title" type="text" id="txt_title" size="40" value="<?php echo $row->title;?>">
        <!-- ---------------------------------------------将公告 ID 的值赋给隐藏域----------------------------------------- -->
        <input name="id" type="hidden"   value="<?php echo $row->id;?>"></td>
     </tr>
     <tr>
         <td height="124" align="center">公告内容：</td>
         <td><textarea name="txt_content"> <?php echo $row->content;?></textarea></td>
     </tr>
     <tr>
         <td height="40" colspan="2" align="center">
         <input name="Submit" type="submit" class="btn_grey" value="修改" onClick="return check(form1);">
         <input type="reset" name="Submit2" value="重置"></td>
      </tr>
    </table>
  </form>
```

（4）提交表单信息到数据处理页 check_modify_ok.php。首先，连接 MySQL 数据库服务器，然后选择数据库，设置 GB2312 的编码格式。应用 POST 方法获取表单信息，接下来应用 mysql_query()函数向 MySQL 数据库服务器发送修改公告信息的 SQL 语句，最后应用 if…else 条件语句对修改后的信息进行判断，并弹出相应的提示信息，重新定位到公告信息编辑页。其代码如下：

```
<?php
/****************连接数据源，读者可将此处封装成独立的页，然后应用 include 语句调用，效率会很高**** ************/
$conn=mysql_connect("localhost","root","111") or die("数据库服务器连接错误".mysql_error());
mysql_select_db("db_database20",$conn) or die("数据库访问错误".mysql_error());
mysql_query("set names gb2312");
/*********************************************************************************** ****************/
$title=$_POST[txt_title];                                    //获取更改的公告主题
$content=$_POST[txt_content];                                //获取更改的公告内容
$id=$_POST[id];                                              //获取更改的公告 ID
//应用 mysql_query()函数向 MySQL 数据库服务器发送修改公告信息的 SQL 语句
$sql=mysql_query("update tb_affiche set title='$title',content='$content' where id=$id");
if($sql){
    echo "<script>alert('公告信息编辑成功！');history.back();window.location.href='modify.php?id=$id';</script>";
}else{
    echo "<script>alert('公告信息编辑失败！');history.back();window.location.href='modify.php?id=$id';</script>";
}
?>
```

（5）在 IE 浏览器中输入地址，按 Enter 键，在页面中输入查询关键字，单击“搜索”按钮，即可输出检索到的公告信息资源，单击✍超链接，弹出编辑公告详细信息页，对指定的公告信息进行编辑，单击“修改”按钮，即可完成指定公告信息的编辑操作，运行结果如图 20.18 所示。



图 20.17　分页显示公告信息的运行结果



图 20.18　编辑公告页面的运行结果

20.3.6　应用 delete 命令动态删除公告信息

视频讲解：光盘\TM\Video\第 20 章\应用 delete 命令动态删除公告信息.exe

公告信息是对一些重要的、新鲜的事务进行管理，因此，为了节省系统资源，可以定期对公告主题及公告内容的信息进行删除。

例 20.14　应用 delete 命令动态编辑数据库中的公告信息，然后通过 mysql_query()函数将 SQL 语句发送到 MySQL 服务器，从而完成数据的删除操作。（**实例位置：光盘\TM\Instance\20\20.14**）

程序开发步骤如下。

（1）在菜单导航页 menu.php 的 image_09.gif 图片上添加热区，链接 delete_affiche.php 文件。

（2）在 delete_affiche.php 页的右侧信息显示区域应用 select 命令检索出全部的公告信息。添加链接文件 check_del_ok.php，并向 check_del_ok.php 页传递公告 ID 的参数值。其代码如下：

```
<a href="check_del_ok.php?id=<?php echo $row->id;?>">
<img src="images/delete.gif" width="22" height="22" border="0">
</a>
```

（3）提交公告信息的 ID 到数据处理页 check_del_ok.php。首先，连接 MySQL 数据库服务器，然后选择数据库，设置 GB2312 的编码格式。应用 GET 方法获取欲删除的公告 ID，接下来应用 mysql_query()函数向 MySQL 数据库服务器发送删除公告信息的 SQL 语句，最后应用 if…else 条件语句对修改后的信息进行判断，并弹出相应的提示信息，重新定位到公告信息编辑页。其代码如下：

```
<?php
$conn=mysql_connect("localhost","root","root") or die("数据库服务器连接错误".mysql_error());
mysql_select_db("db_database20",$conn) or die("数据库访问错误".mysql_error());
mysql_query("set names gb2312");
$id=$_GET[id];
$sql=mysql_query("delete from tb_affiche where id=$id");
if($sql){
    echo "<script>alert('公告信息删除成功！');history.back();window.location.href='delete_affiche.php?id=$id';
</script>";
}else{
    echo "<script>alert('公告信息删除失败！');history.back();window.location.href='delete_affiche.php?id=$id';
</script>";
}
?>
```

技巧 由于数据处理页 check_del_ok.php 中都是动态代码，没有指定编码格式为 GB2312 编码类型，以致读者在用 Dreamweaver 开发工具打开该文件时，中文部分将会显示乱码。为了解决这一问题，读者可以在该页指定其编码格式。其代码如下：

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
```

（4）在 IE 浏览器中输入地址，按 Enter 键，在页面中输入查询关键字，单击“搜索”按钮，即可输出检索到的公告信息资源。单击指定公告信息后面的按钮，弹出删除公告信息提示，然后单击“确定”按钮，即可完成对指定公告信息的删除操作，运行结果如图 20.19 所示。



图 20.19　删除公告页面信息的运行结果

20.4　PHP 操作 MySQL 事务

事务处理机制在程序开发过程中有着非常重要的作用，它可以使整个系统更加安全。例如，在银行处



理转账业务时，如果 A 账户中的金额刚被发出，而 B 账户还没来得及接受就发生停电，会给银行和个人带来很大的经济损失。采用事务处理机制，一旦在转账过程中发生意外，则程序将回滚，不做任何处理。

下面讲解使用事务处理技术实现关联表信息的删除方法。

例 20.15　将采用事务处理方式，对学生信息表和学生成绩表中的数据进行删除。运行本实例，学生信息表及学生成绩表信息分别如图 20.20 和图 20.21 所示。当删除图 20.20 中的学生信息后，查看学生成绩信息时可以发现，与该学生对应的成绩也被全部删除。**（实例位置：光盘\TM\Instance\20\20.15）**

程序开发步骤如下。

（1）建立数据库及数据表并实现与数据的连接。其代码如下：

```
<?php
$conn=new mysqli("localhost","root","111","db_database20");
$conn->query("set names utf8");
?>
```

（2）显示所有学生的基本信息。其代码如下：

```
<?php
      include_once("conn.php");
     $sql=$conn->query("select * from tb_stu");
     $info=$sql->fetch_array(MYSQLI_ASSOC);
     if($info==NULL)
      {
        echo "暂无学生信息!";
      }
      else
      {
        do
         {
?>
   <tr>
        <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info[sname];?></div></td>
        <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info[sno];?></div></td>
        <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info[sage];?></div></td>
        <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info[saddress];?></div></td>
        <td height="25" bgcolor="#FFFFFF"><div align="center"><?php echo $info[ssfzh];?></div></td>
        <td bgcolor="#FFFFFF"><div align="center"><a href="javascript:if(window.confirm('确定删除该学生信
息么?')==true){window.location.href='delete.php?id=<?php echo $info[id];?>';}">删除</a></div></td>
      </tr>
 <?php
   }
     while($info=$sql->fetch_array(MYSQLI_ASSOC));
   }
 ?>
```

在实现该模块功能时，可以利用 JavaScript 实现该页与 delete.php 页面的信息传递。其代码如下：

```
<a href="javascript:if(window.confirm('确定删除该学生信息嘛')==true)
{window.location.href='delete.php?id=<?php echo $info[id];?>';}">删除
</a>
```

window 对象的 confirm()方法用于弹出一个对话框，提示用户真正删除某学生的所有信息。

（3）利用事务处理机制对学生的基本信息进行删除。其代码如下：

```
<?php
$id=$_GET[id];
include_once("conn.php");
$conn->autocommit(false);
if(!$conn->query("delete from tb_sco where id='".$id."'"))
{
```

```
 $conn->rollback();
}
if(!$conn->query("delete from tb_stu where id='".$id."'"))
{
 $conn->rollback();
}
$conn->commit();
$conn->autocommit(true);
echo "<script>window.location.href='index.php';</script>";
?>
```

学生成绩管理系统

查看学生信息　查看学生成绩

学生姓名	学号	年龄	住址	身份证号	操作
lzh	0312317	18	吉林省长春市	22010245****	删除
lsh	0312316	17	吉林省吉林市	20211032855******	删除

图 20.20　查看学生信息

学生成绩管理系统

查看学生信息　查看学生成绩

学生姓名	学号	语文	数学	外语
lzh	0312317	85	98	78
lbh	0312316	75	98	86
lbh	0312316	79	95	80
lzh	0312317	78	82	89

图 20.21　查看学生成绩

20.5　PHP 操作 MySQL 存储过程

MySQL 5.0 以后的版本开始支持存储过程，存储过程具有一致性、高效性、安全性和体系结构等特点，本节将具体讲解 PHP 是如何操纵 MySQL 存储过程的。

下面介绍如何使用存储过程实现用户登录。

例 20.16　使用 MySQL 存储过程实现用户登录身份的验证。运行本实例，如图 20.22 所示。在图中登录窗口的文本框中输入用户名和密码，然后单击“登录”按钮，如果用户登录的用户名和密码正确，则会将页面转向如图 20.23 所示的登录成功提示页面。（**实例位置：光盘\TM\Instance\20\20.16**）

程序开发步骤如下。

（1）创建存储过程，通过 query()方法调用存储过程。其代码如下：

```
delimiter //
create procedure pro_login1(in un varchar(50), in pass varchar(50))
begin
select * from tb_login where username=un and password=pass;
end
//
```

（2）通过 MySQLi 扩展库建立与 MySQL 数据库的连接，并设置数据库字符集为 utf-8。其代码如下：

```
<?php
$conn=new mysqli("localhost","root","111","db_database20");
$conn->query("set names utf8");
?>
```

（3）建立用户登录表单，当用户在表单中输入用户名和密码，并单击“提交”按钮后，通过如下代码验证用户的登录信息是否正确。

```
<?php
if(isset($_POST['username']) && trim($_POST['username'])!='')
{
    require_once 'Db.php';
    $username = trim($_POST['username']);
    $password = trim($_POST['password']);
    $sql = $mysqli->query("call pro_login1('".$username."', '".$password."')");
```

```
	$info = $sql->fetch_array(MYSQLI_ASSOC);
	if($info != null){
		$_SESSION['loginUsername'] =   $username;
		echo '<script>window.location.href="success.php";</script>';
	}else {
		echo '<div style="width:300px; height:30px; line-height:30px; border:1px solid #E59B04; background-color:#FCF2E0; color:#FF0000;">用户名或密码输入有误</div>';
	}
}
?>
```

上述代码中，首先判断$_POST['username']的值是否被设置，则首先使用 require_once 语句包含 conn.php 文件，然后使用 MySQLi 扩展库的 query()方法执行存储过程 pro_login，并使用 query()方法返回的对象调用 fetch_array()获得结果集，最后通过判断结果集是否为 null 来判断用户所输入的信息是否正确。



图 20.22　用户登录窗口

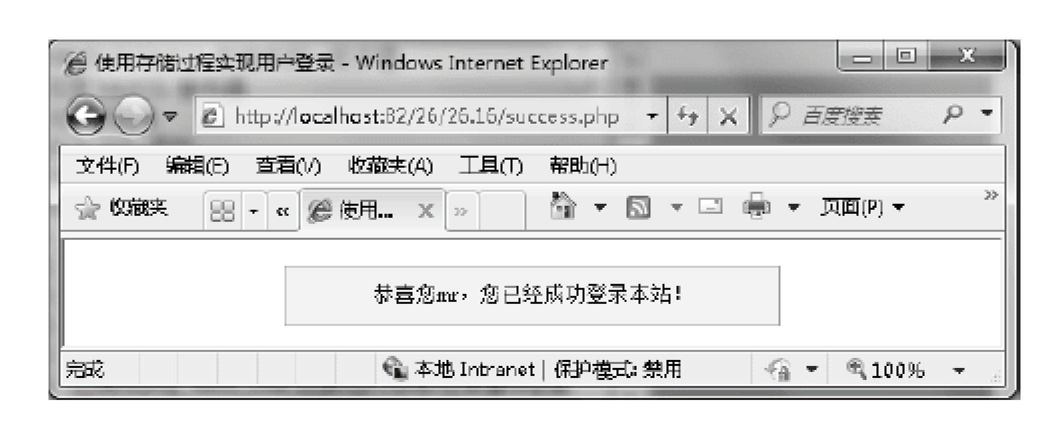


图 20.23　登录成功的提示信息

20.6　实　　战

视频讲解：光盘\TM\Video\第 20 章\实战.exe

MySQL 数据库函数的使用方法是每位 PHP 开发人员所必须掌握的，只有掌握了这些函数的使用方法才能具备最基本数据库系统开发的能力。为了巩固本章所讲的内容，下面通过具体实例进一步讲解和深化 MySQL 数据库函数的应用及 PHP 操纵 MySQL 数据库进行增、删、改、查的操作方式。

20.6.1　输入页码跳转到指定页

在网站的应用过程中，经常需要将数据库中很多记录显示到页面中来，如果将所有的记录在一个页面中显示，会给浏览者带来很多麻烦，最好的解决方法就是使用分页技术，将记录进行分页显示。通过分页读取记录信息，在每页中显示几条记录，这样既方便浏览者的浏览，也节省了页面的空间。

例 20.17　以分页的方式显示员工信息，并通过输入页码跳转到指定的页。**（实例位置：光盘\TM\Instance\20\20.17）**

程序开发步骤如下。

（1）首先建立与 MySQL 数据库的连接，并选择数据库 db_database20，如果发生错误则使用 mysql_error()函数输出错误号，最后使用 mysql_query()函数执行 set names gb2312 命令来设置数据库的字符集为简体中文。其代码如下：

```
<?php
$id=mysql_connect("localhost","root","111")or dir('连接失败:' . mysql_error());
if(mysql_select_db("db_database20",$id))
```

```
	echo "";
else
	echo ('连接失败:' . mysql_error());
mysql_query("set names gb2312");
?>
```

（2）使用分页的方式显示员工信息。其代码如下：

```
<?php
include("conn/conn.php");
if ($page=="") {$page=1;};if ($ljjl=="") {
	$ljjl=0;
};
?>
<body>
<table width="450" border="1" cellpadding="0" cellspacing="0" bgcolor="#4DB1FD">
	<tr>
		<td width="100" height="25" align="center" class="STYLE2">姓名</td>
		<td width="100" align="center" class="STYLE2">编号</td>
		<td width="125" align="center" class="STYLE2">电话</td>
		<td width="175" align="center" class="STYLE2">地址</td>
	 </tr>
<?php
if($page){
	$page_size=2;                                                    //每页显示两条记录
	$query="select count(*) as total from tb_insert where id";      //从数据库中读取数据
	$result=mysql_query($query);
	$message_count=mysql_result($result,0,"total");                 //获取总的记录数
	$page_count=ceil($message_count/$page_size);                    //获取总的页数
	$offset=($page-1)*$page_size;
	$query="select * from tb_insert where id order by id desc limit $offset, $page_size";
	$result=mysql_query($query);
	while ($myrow=@mysql_fetch_array($result)){
?>
<tr>
	<td height="25" align="center"><span class="STYLE2"><?php echo $myrow[name];?></span></td>
	<td align="center"><span class="STYLE2"><?php echo $myrow[number];?></span></td>
	<td align="center"><span class="STYLE2"><?php echo $myrow[tel];?></span></td>
	<td align="center"><span class="STYLE2"><?php echo $myrow[address];?></span></td>
</tr>
<?php
	}
}
?>
</table>
<table width="450" border="1" cellpadding="0" cellspacing="0" bgcolor="#4DB1FD">
	<tr>
		<td width="45%" align="center"><span class="STYLE1">  页次：
			<?php echo $page;?>/ <?php echo $page_count;?> 页 记录：
			<?php echo $message_count;?> 条  </span>
		</td>
		<td width="55%" height="22" align="center"><span class="STYLE1"> 分页：
<?php
	if($page!=1){
		echo   "<a href=index.php?page=1>首页</a> ";
```

```
            echo "<a href=index.php?page=".($page-1).">上一页</a> ";
        }
        if($page<$page_count){
             echo "<a href=index.php?page=".($page+1).">下一页</a> ";
             echo   "<a href=index.php?page=".$page_count.">尾页</a>";
        }
?>
            </span></td>
        </tr>
</table>
```

运行上述实例，结果如图 20.24 所示，在图中的文本框中输入要浏览的页码，然后单击“跳转”按钮即可跳转到要浏览的页面。

姓名	编号	电话	地址
独孤**	1007	45612312	大理
欧阳*	1006	12345678	通化

页次：1 / 3 页 记录：6 条	分页： 下一页 尾页 跳转

图 20.24　运行结果

20.6.2　图片的分栏分页显示

PHP 实现 MySQL 数据库中的数据分页显示相对简单，在此基础上还可以实现如图片、代码块的分栏分页显示。

例 20.18　图片的分栏分页显示。（**实例位置：光盘\TM\Instance\20\20.18**）

程序开发步骤如下：

（1）创建数据库连接。其代码如下：

```
<?php
$id=mysql_connect("localhost","root","111")or die('连接失败:' . mysql_error());
if(mysql_select_db("db_database20",$id))
echo "";
else
echo ('连接失败:' . mysql_error());
mysql_query("set names gb2312");
?>
```

（2）通过分页、分类、分栏方法将数据库中的数据显示到页面中。其代码如下：

```
<table width="400" border="0" cellpadding="0" cellspacing="1" bgcolor="#663300">
<?php
    if($page){
        $page_size=8;
        $query="select count(*) as total from tb_picture where sort='1'";
        $result=mysql_query($query);
        $message_count=mysql_result($result,0,"total");
        $page_count=ceil($message_count/$page_size);
        $offset=($page-1)*$page_size;
        $query="select * from tb_picture where sort='1' limit $offset, $page_size";
        $result=mysql_query($query);
        $row=mysql_fetch_array($result);
        if($row[sort]==1){
            for($i=1; $i<=2; $i++) {
                echo "<tr>";
                if($i==1) {
                    $query="select * from tb_picture where sort='1' limit $offset, $page_size";
                    $result=mysql_query($query); }
                    $j=1;
                while($myrow=mysql_fetch_array($result)){
```

```
                    $id=$myrow[id];
                    if ($j<=4){
          ?>
           <td height="50" align="center" valign="top" bgcolor="#FFFFFF" class="STYLE1">
     <table width="50" border="0" cellspacing="0">
          <tr align="center" valign="middle">
              <td  width="50"  height="50"  align="center"  valign="middle"  background="images/beijing.jpg">
<?php echo $myrow[id];?><img src="image_1.php?recid=<?php echo $myrow[id];?>"></td>
           </tr>
    </table></td>
<?php
       }
        ++$j;
        if($j==5){
            break;
        }
}
echo "</tr>";
    }
}
?>
<?php
for($i=1; $i<=2; $i++) {
    echo "<tr>";
?>
<?php
if($i==1){
     $query="select * from tb_picture where sort='2' limit $offset, $page_size";
     $result=mysql_query($query); }
     $j=1;
     while($myrow=mysql_fetch_array($result)){
     $id=$myrow[id];
     if($j<=4){
?>
              <td  height="50"  align="center"  valign="top"  bgcolor="#FFFFFF"  class="STYLE1">  <table
width="50" border="0" cellspacing="0">
          <tr align="center" valign="middle">
               <td  width="50"  height="50"  align="center"  valign="middle"  background="images/beijing.jpg">
<?php echo $myrow[id];?><img src="image_1.php?recid=<?php echo $myrow[id];?>"></td>
           </tr>
</table>
</td>
  <?php
             }
             ++$j;
             if ($j==5){   break;
         }
    }
    echo "</tr>";
}
?>
</table>
     <table width="400" border="0" cellspacing="0" cellpadding="0">
```

```
        <tr>
          <td width="50%" height="22" class="hongse01 STYLE1">  页次：<font class="huise01">
<?php echo $page;?></font>
                    / <font class="huise01"><?php echo $page_count;?> </font>页 记录：<font class="huise01">
<?php echo $message_count;?>
                    </font>条  </td>
                  <td width="39%" height="22" class="hongse01 STYLE1"> 分页：
      <?php
                $xsoudh="id=$id";
                $next=$ljjl*10;
                $n=$ljjl-1;
                $m=$ljjl+1;
                $prev_page=$page-10;
          if ($ljjl==0){
            echo "<img src=\"images/02.jpg\" width=\"8\" height=\"9\" title=\"首页\">";
          }else{
            echo  "<a  href='$PATH_INFO?page=1'><img  src=\"images/02.jpg\"  width=\"8\"  height=\"9\"
border=\"0\" title=\"首页\"></a> ";
          $ccc=$vv-10;
            echo  "<a  href='$PATH_INFO?page=$prev_page'><img  src=\"images/01.jpg\"  width=\"8\"
height=\"9\" title=\"上十页\"></a>";
          }
      ?>
<?php
     for($j=1;$j<=$page_count;$j++){
          $pnext=$next+$j;
          if($mm==10){ break; }
          if($mm>$page_count){ break; }
          if($page_count-$vv<10){
          if ($mm>=$page_count-$vv){ break; }
     }
?>
<?php
          echo "<a href='$PATH_INFO?page=$pnext'> $pnext </a>";
          $mm=$mm+1;
}
?>
 </td>
      <td width="11%"> <span class="STYLE1">
      <?php
          $vv=$vv+$mm;
               if($page_count-$vv<=0){
                    echo "<img src=\"images/03.jpg\" width=\"8\" height=\"9\" title=\"尾页\">";
               }else{
                    echo  "<a  href='$PATH_INFO?page=$pnext'><img  src=\"images/03.jpg\"  width=\"8\"
height=\"9\" title=\"下十页\"></a>";
               }
        if ($message_count==0){ echo "没有记录!"; }
     }
?>
     </span></td></tr>
</table>
```

（3）读取数据库中二进制文件调用的 image_1.php 文件。其代码如下：

```
<?php
$id=mysql_connect('localhost','root','111');
mysql_select_db('db_database20',$id);
$query="select * from tb_picture where id=".$recid;
$result=mysql_query($query);
if(!$result) die("error: mysql query");
    $num=mysql_num_rows($result);
if($num<1) die("error: no this recorder");
    $data = mysql_result($result,0,"xq");
mysql_close($id);
echo $data;
?>
```

运行本实例，如图 20.25 所示，通过页面中的分页链接即可以分栏分页的方式浏览所有图片。

20.6.3　查询图书信息表中的前 3 条记录

例 20.19　查询图书信息表 tb_book 中的前三条图书信息记录，单击“查询”按钮即可查询前三条图书信息记录，运行结果如图 20.26 所示。（**实例位置：光盘\TM\Instance\20\20.19**）

具体实现步骤如下。

（1）创建数据库连接文件 conn.php。其代码如下：

```
<?php
$conn=mysql_connect("localhost","root","111");                    //连接数据库服务器
mysql_select_db("db_database20,$conn);                            //连接 db_database20 数据库
mysql_query("set names utf8");                                    //设置数据库编码格式
?>
```

（2）创建 index.php 文件，设计图书信息查询表单页面，效果如图 20.26 所示。

（3）完成查询操作。其代码如下：

```
<?php
include("conn.php");                                              //包含 conn.php 文件
if(isset($_POST['Submit']) and $_POST['Submit']=="查询"){          //判断用户是否执行查询操作
     $select=mysql_query("select * from tb_book1   where id limit 3 ",$conn);  //查询数据库中前三条记录
} else{
     $select=mysql_query("select * from tb_book1",$conn);         //查询所有记录
}
?>
<?php
     while($rows=mysql_fetch_array($select)){                     //使用 while 循环查询结果
?>
          <tr>
           <td><?php echo $rows['id'];?> </td>               <!--输出 id-->
           <td><?php echo $rows['bookname'];?> </td>         <!--输出书名-->
           <td><?php echo $rows['maker'];?> </td>            <!--输出作者-->
           <td><?php echo $rows['publisher'];?> </td>        <!--输出出版社-->
           <td><?php echo $rows['issuDate'];?> </td>         <!--输出出版时间-->
           <td><?php echo $rows['price'];?> </td>            <!--输出价格-->
         </tr>
<?php
     }                                                            //结束循环
?>
```

运行结果如图 20.26 所示。



1	2	3	4
5	6	7	8
13	14	15	16
17	18	19	20

页次：1 / 2 页 记录：12 条　　分页：⏮ 1 2 ⏭

图 20.25　图片的分栏分页显示



图 20.26　查询前三条图书信息记录

20.6.4　对查询结果进行降序排列输出

例 20.20　对图书信息表 tb_news 中的图书信息记录按图书序号进行降序排列，单击“降序排列”按钮，即可按图书序号进行降序排列。**（实例位置：光盘\TM\Instance\20\20.20）**

具体步骤如下。

（1）创建数据库连接文件 conn.php。其代码如下：

```
<?php
$conn=mysql_connect("localhost","root","111");                  //连接数据库服务器
mysql_select_db("db_database20",$conn);                         //连接 db_database20 数据库
mysql_query("set names utf8");                                  //设置数据库编码格式
?>
```

（2）创建 index.php 文件，设计图书信息查询表单页面，效果如图 20.27 所示。

（3）完成降序排列操作。其代码如下：

```
<?php
include("conn.php");                                            //包含 conn.php 文件
if(!isset($_POST['Submit2'])){                                  //判断按钮事件为空时
    $select=mysql_query("select * from tb_book1",$conn);        //显示数据表中所有记录
    $rows=mysql_fetch_array($select);
}else{                                                          //否则按 id 进行降序排列，并获取返回值
    $select=mysql_query("select * from tb_book1 order by id desc",$conn);
    $rows=mysql_fetch_array($select);
}
?>

<?php
    do{                                                         //使用 do while 循环
?>
      <tr>
          <td><?php echo $rows['id'];?> </td>              <!--输出 id-->
          <td><?php echo $rows['bookname'];?> </td>        <!--输出书名-->
          <td><?php echo $rows['maker'];?> </td>           <!--输出作者-->
          <td><?php echo $rows['publisher'];?> </td>       <!--输出出版社-->
          <td><?php echo $rows['issuDate'];?> </td>        <!--输出出版时间-->
          <td><?php echo $rows['price'];?> </td>           <!--输出价格-->
        </tr>
<?php
```

```
	}while($rows=mysql_fetch_array($select));				//执行降序排列操作
?>
```

运行结果如图 20.27 所示。

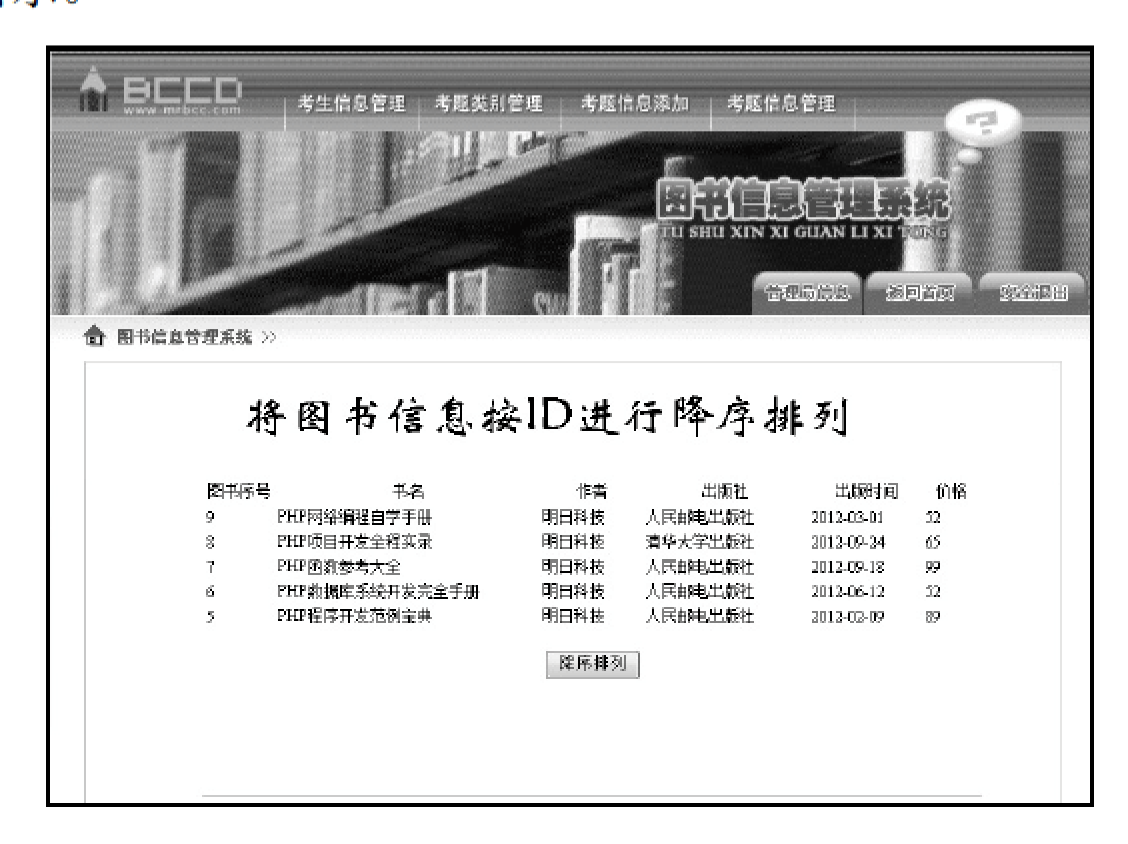


图 20.27　将图书信息按 ID 降序排列

20.7　小　　结

本章首先介绍了 PHP 访问 MySQL 数据库的一般流程，然后详细介绍了该流程每一步骤的具体实现方法，并重点讲解了常用函数的使用方法，最后通过公告栏实例更深层次地介绍了 PHP 如何实现对 MySQL 数据库进行增、删、改、查操作。通过本章的学习，读者能够掌握 PHP 操作 MySQL 数据库的一般流程，掌握常用 MySQL 函数的使用方法，并能够具备独立完成基本数据库程序的能力。希望本章能够起到抛砖引玉的作用，帮助读者在此基础上更深层次地学习 PHP 操作 MySQL 数据库的相关技术，并进一步学习使用面向对象的方式操作 MySQL 数据库的方法。

20.8　学习成果检验

1．查询结果分页显示，运行效果如图 20.28 所示。（**实例位置：光盘\TM\Instance\20\20.21**）

2．按照员工编号查询员工的详细信息，运行效果如图 20.29 所示。（**实例位置：光盘\TM\Instance\20\20.22**）

姓名	编号	电话	地址
刘**	1005	87456123	四平
潘**	1004	12345678	长春
页次：3 / 3 页 记录：6 条		分页： 首页 上一页	

图 20.28　查询结果分页显示



图 20.29　按编号查询员工信息

3．动态显示新闻信息，截取部分新闻主题字符串，屏蔽乱码。（**实例位置：光盘\TM\Instance\20\20.23**）



第21章

PDO 数据库抽象层

（视频讲解：60 分钟）

在 PHP 的早期版本中，各种不同的数据库扩展（MySQL、MS SQL、Oracle）根本没有真正的一致性，虽然都可以实现相同的功能，但是这些扩展却互不兼容，都有各自的操作函数，各自为政。结果导致 PHP 的维护非常困难，可移植性也非常差，为了解决这些问题，PHP 的开发人员编写了一种轻型、便利的 API 来统一各种数据库的共性，从而达到 PHP 脚本最大程度的抽象性和兼容性，这就是数据库抽象层。而在本章中将要介绍的是目前 PHP 抽象层中最为流行的一种——PDO 抽象层。

通过阅读本章内容，你可以：

- 了解什么是 PDO
- 掌握 PDO 连接数据库
- 掌握在 PDO 中执行 SQL 语句
- 掌握在 PDO 中获取结果集
- 掌握在 PDO 中捕获 SQL 语句中的错误
- 掌握 PDO 中的错误处理
- 掌握 PDO 中的事务处理
- 掌握 PDO 中的存储过程

21.1　什么是 PDO

21.1.1　PDO 概述

PDO 是 PHP Date Object（PHP 数据对象）的简称，它是与 PHP 5.1 版本一起发行的，目前支持的数据库包括 Firebird、FreeTDS、Interbase、MySQL、MS SQL Server、ODBC、Oracle、Postgre SQL、SQLite 和 Sybase。有了 PDO，不必再使用 mysql_*函数、oci_*函数或者 mssql_*函数，也不必再为它们封装数据库操作类，只需要使用 PDO 接口中的方法就可以对数据库进行操作。在选择不同的数据库时，只需修改 PDO 的 DSN（数据源名称）。

在 PHP 6 中将默认使用 PDO 连接数据库，所有非 PDO 扩展将会在 PHP 6 中被移除。该扩展提供 PHP 内置类 PDO 来对数据库进行访问，不同数据库使用相同的方法名，来解决数据库连接不统一的问题。

21.1.2　PDO 特点

PDO 是一个“数据库访问抽象层”，作用是统一各种数据库的访问接口，与 mysql 和 mssql 函数库相比，PDO 让跨数据库的使用更具有亲和力；与 ADODB 和 MDB2 相比，PDO 更高效。

PDO 将通过一种轻型、清晰、方便的函数，统一各种不同 RDBMS 库的共有特性，实现 PHP 脚本最大程度的抽象性和兼容性。

PDO 吸取现有数据库扩展成功和失败的经验教训，利用 PHP 5 的最新特性，可以轻松地与各种数据库进行交互。

PDO 扩展是模块化的，能够在运行时为你的数据库后端加载驱动程序，而不必重新编译或重新安装整个 PHP 程序。例如，PDO_MySQL 扩展会替代 PDO 扩展实现 MySQL 数据库 API。还有一些用于 Oracle、PostgreSQL、ODBC 和 Firebird 的驱动程序，更多的驱动程序尚在开发中。

21.1.3　安装 PDO

PDO 是与 PHP 5.1 一起发行的，默认包含在 PHP 5.1 中。由于 PDO 需要 PHP 5 核心面向对象特性的支持，因此其无法在 PHP 5.0 之前的版本中使用。

默认情况下，PDO 在 PHP 5.2 中为开启状态，但是要启用对某个数据库驱动程序的支持，仍需要进行相应的配置操作。

在 Linux 环境下，要使用 MySQL 数据库，可以在 configure 命令中添加如下选项：

```
--with-pdo-mysql=/path/to/mysql/installation
```

在 Windows 环境下，PDO 在 php.ini 文件中进行配置，如图 21.1 所示。

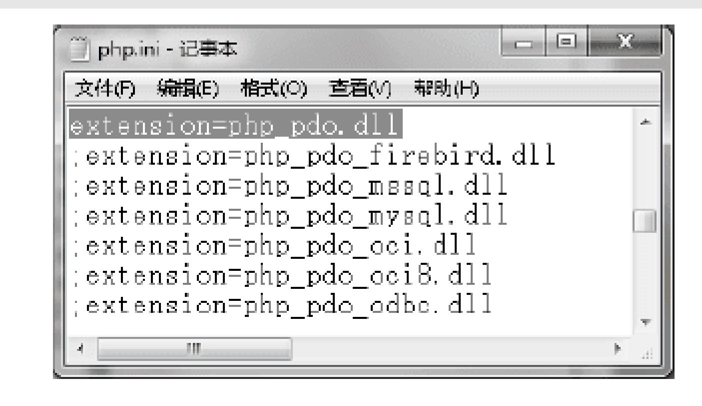
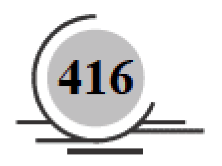


图 21.1　Windows 环境下配置 PDO

要启用 PDO，首先必须加载“extension=php_pdo.dll”，如果要想其支持某个具体的数据库，那么还要加载对应的数据库选项。例如，要支持 MySQL 数据库，则需要加载“extension=php_pdo_mysql.dll”选项。





在完成数据库的加载后，要保存 php.ini 文件，并且重新启动 Apache 服务器，修改即可生效。

21.2　PDO 连接数据库

视频讲解：光盘\TM\Video\第 21 章\PDO 连接数据库.exe

21.2.1　PDO 构造函数

在 PDO 中，要建立与数据库的连接需要实例化 PDO 的构造函数。PDO 构造函数的语法如下：

```
__construct(string $dsn[,string $username[,string $password[,array $driver_options]]])
```

参数说明如下。

- ☑ dsn：数据源名，包括主机名端口号和数据库名称。
- ☑ username：连接数据库的用户名。
- ☑ password：连接数据库的密码。
- ☑ driver_options：连接数据库的其他选项。

通过 PDO 连接 MySQL 数据库的代码如下：

```
<?php
	header("Content-Type:text/html;charset=utf-8");			//设置页面的编码格式
	$dbms='mysql';							//数据库类型
	$dbName='db_database21';					//使用的数据库名称
	$user='root';							//使用的数据库用户名
	$pwd='111';							//使用的数据库密码
	$host='localhost';						//使用的主机名称
	$dsn="$dbms:host=$host;dbname=$dbName";
	try {								//捕获异常
		$pdo=new PDO($dsn,$user,$pwd);				//实例化对象
		echo "PDO 连接 MySQL 成功";
	} catch (Exception $e) {
		echo $e->getMessage()."<br>";
	}
?>
```

21.2.2　DSN 详解

DSN 是 Data Source Name（数据源名称）的首字母缩写。它提供连接数据库需要的信息。PDO 的 DSN 包括 3 部分：PDO 驱动名称（如 mysql、sqlite 或者 pgsql）、冒号和驱动特定的语法。每种数据库都有其特定的驱动语法。

在使用不同的数据库时，必须明确数据库服务器是完全独立于 PHP 的实体。虽然笔者在讲解本书的内容时，数据库服务器和 Web 服务器是在同一台计算机上，但是实际的情况可能不是如此。数据库服务器可能与 Web 服务器不是在同一台计算机上，此时要通过 PDO 连接数据库时，就需要修改 DSN 中的主机名称。

由于数据库服务器只在特定的端口上监听连接请求。而每种数据库服务器具有一个默认的端口号（MySQL 是 3306），数据库管理员又可以对端口号进行修改，因此有可能 PHP 找不到数据库的端口，此

时就可以在 DSN 中包含端口号。

另外由于一个数据库服务器中可能拥有多个数据库，所以在通过 DSN 连接数据库时，通常都包括数据库名称，这样可以确保连接的是你想要的数据库，而不是其他的数据库。

21.3 PDO 中执行 SQL 语句

视频讲解：光盘\TM\Video\第 21 章\PDO 中执行 SQL 语句.exe

在 PDO 中，可以使用下面的 3 种方法来执行 SQL 语句。

21.3.1 exec()方法

exec()方法返回执行后受影响的行数。其语法如下：

```
int PDO::exec ( string statement )
```

参数 statement 是要执行的 SQL 语句。

该方法返回执行查询时受影响的行数，通常用于 INSERT、DELETE 和 UPDATE 语句中。

21.3.2 query()方法

query()方法通过用于返回执行查询后的结果集。其语法如下：

```
PDOStatement PDO::query ( string statement )
```

参数 statement 是要执行的 SQL 语句。它返回的是一个 PDOStatement 对象。

21.3.3 预处理语句——prepare()和 execute()

预处理语句包括 prepare()和 execute()两个方法。首先，通过 prepare()方法做查询的准备工作，然后，通过 execute()方法执行查询。并且还可以通过 bindParam()方法来绑定参数提供给 execute()方法。其语法如下：

```
PDOStatement PDO::prepare ( string statement [, array driver_options] )
bool PDOStatement::execute ( [array input_parameters] )
```

21.4 PDO 中获取结果集

视频讲解：光盘\TM\Video\第 21 章\PDO 中获取结果集.exe

在 PDO 中获取结果集有 3 种方法：fetch()、fetchAll()和 fetchColumn()。

21.4.1 fetch()方法

fetch()方法获取结果集中的下一行。其语法格式如下：

```
mixed PDOStatement::fetch ( [int fetch_style [, int cursor_orientation [, int cursor_offset]]] )
```

参数说明如下：

☑ fetch_style：控制结果集的返回方式，其可选方式如表 21.1 所示。

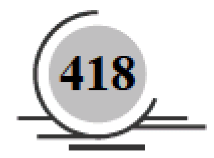


表 21.1　fetch_style 控制结果集的可选值

值	说　明
PDO::FETCH_ASSOC	关联数组形式
PDO::FETCH_NUM	数字索引数组形式
PDO::FETCH_BOTH	两者的数组形式都有，这是默认的
PDO::FETCH_OBJ	按照对象的形式，类似于以前的 mysql_fetch_object()
PDO::FETCH_BOUND	以布尔值的形式返回结果，同时将获取的列值赋给 bindParam()方法中指定的变量
PDO::FETCH_LAZY	以关联数组、数字索引数组和对象 3 种形式返回结果

☑　cursor_orientation：PDOStatement 对象的一个滚动游标，可用于获取指定的一行。

☑　cursor_offset：游标的偏移量。

例 21.1　通过 fetch()方法获取结果集中下一行的数据，进而应用 while 语句完成数据库中数据的循环输出。**（实例位置：光盘\TM\Instance\21\21.1）**

创建 index.php 文件，设计网页页面。首先，通过 PDO 连接 MySQL 数据库。然后，定义 SELECT 查询语句，应用 prepare()和 execute()方法执行查询操作。接着，通过 fetch()方法返回结果集中下一行数据，同时设置结果集以关联数组形式返回。最后，通过 while 语句完成数据的循环输出。其关键代码如下：

```
<?php
$dbms='mysql';       //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
$host='localhost';                              //数据库主机名
$dbName='db_database21';                        //使用的数据库
$user='root';                                   //数据库连接用户名
$pass='111';                                    //对应的密码
$dsn="$dbms:host=$host;dbname=$dbName";
try {
    $pdo = new PDO($dsn, $user, $pass);         //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
     $query="select * from tb_pdo_mysql";       //定义 SQL 语句
     $result=$pdo->prepare($query);             //准备查询语句
     $result->execute();                        //执行查询语句，并返回结果集
     while($res=$result->fetch(PDO::FETCH_ASSOC)){  //循环输出查询结果集，并且设置结果集的为关联索引
     ?>
          <tr>
            <td height="22" align="center" valign="middle"><?php echo $res['id'];?></td>
            <td align="center" valign="middle"><?php echo $res['pdo_type'];?></td>
            <td align="center" valign="middle"><?php echo $res['database_name'];?></td>
            <td align="center" valign="middle"><?php echo $res['dates'];?></td>
            <td align="center" valign="middle"><a href="#">删除</a></td>
          </tr>
<?php
     }
            } catch (PDOException $e) {
    die ("Error!: " . $e->getMessage() . "<br/>");
}
            ?>
```

运行结果如图 21.2 所示。

21.4.2　fetchAll()方法

fetchAll()方法获取结果集中的所有行。其语法如下：

```
array PDOStatement::fetchAll ( [int fetch_style [, int column_index]] )
```

参数说明如下：

☑ fetch_style：控制结果集中数据的显示方式。

☑ column_index：字段的索引。

其返回值是一个包含结果集中所有数据的二维数组。

例 21.2 通过 fetchAll()方法获取结果集中所有行，并且通过 for 语句读取二维数组中的数据，完成数据库中数据的循环输出。**（实例位置：光盘\TM\Instance\21\21.2）**

创建 index.php 文件，设计网页页面。首先，通过 PDO 连接 MySQL 数据库。然后，定义 SELECT 查询语句，应用 prepare()和 execute()方法执行查询操作。接着，通过 fetchAll()方法返回结果集中所有行。最后，通过 for 语句完成结果集中所有数据的循环输出。其关键代码如下：

```
<?php
$dbms='mysql';        //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
$host='localhost';                                  //数据库主机名
$dbName='db_database21';                            //使用的数据库
$user='root';                                       //数据库连接用户名
$pass='111';                                        //对应的密码
$dsn="$dbms:host=$host;dbname=$dbName";
try {
    $pdo = new PDO($dsn, $user, $pass);             //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
     $query="select * from tb_pdo_mysql";           //定义 SQL 语句
     $result=$pdo->prepare($query);                 //准备查询语句
     $result->execute();                            //执行查询语句，并返回结果集
     $res=$result->fetchAll(PDO::FETCH_ASSOC);          //获取结果集中的所有数据
     for($i=0;$i<count($res);$i++){                     //循环读取二维数组中的数据
     ?>
          <tr>
               <td height="22" align="center" valign="middle"><?php echo $res[$i]['id'];?></td>
               <td align="center" valign="middle"><?php echo $res[$i]['pdo_type'];?></td>
               <td align="center" valign="middle"><?php echo $res[$i]['database_name'];?></td>
               <td align="center" valign="middle"><?php echo $res[$i]['dates'];?></td>
               <td align="center" valign="middle"><a href="#">删除</a></td>
          </tr>
<?php
     }
} catch (PDOException $e) {
    die ("Error!: " . $e->getMessage() . "<br/>");
} ?>
```

运行结果如图 21.3 所示。



图 21.2 fetch()方法获取查询结果集中下一行的数据



图 21.3 fetchAll()方法返回结果集中所有数据



21.4.3　fetchColumn()方法

fetchColumn()方法获取结果集中下一行指定列的值。其语法如下：

```
string PDOStatement::fetchColumn ( [int column_number] )
```

可选参数 column_number 设置行中列的索引值，该值从 0 开始。如果省略该参数则将从第 1 列开始取值。

通过 fetchColumn()方法获取结果集中下一行中指定列的值，注意，这里是“结果集中下一行中指定列的值”。

例 21.3　本实例输出数据表中第一列的值，即输出数据的 ID。**（实例位置：光盘\TM\Instance\21\21.3）**

创建 index.php 文件，设计网页页面。首先，通过 PDO 连接 MySQL 数据库。然后，定义 SELECT 查询语句，应用 prepare()和 execute()方法执行查询操作。接着，通过 fetchColumn()方法输出结果集中下一行第一列的值。其关键代码如下。

```
<?php
$dbms='mysql';        //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
$host='localhost';                                    //数据库主机名
$dbName='db_database21';                              //使用的数据库
$user='root';                                         //数据库连接用户名
$pass='111';                                          //对应的密码
$dsn="$dbms:host=$host;dbname=$dbName";
try {
    $pdo = new PDO($dsn, $user, $pass);               //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
        $query="select * from tb_pdo_mysql";          //定义 SQL 语句
        $result=$pdo->prepare($query);                //准备查询语句
        $result->execute();                           //执行查询语句，并返回结果集
    ?>
          <tr>
            <td height="22" align="center" valign="middle"><?php echo $result->fetchColumn(0);?></td>
          </tr>
          <tr>
            <td height="22" align="center" valign="middle"><?php echo $result->fetchColumn(0);?></td>
          </tr>
          <tr>
            <td height="22" align="center" valign="middle"><?php echo $result->fetchColumn(0);?></td>
          </tr>
          <tr>
            <td height="22" align="center" valign="middle"><?php echo $result->fetchColumn(0);?></td>
          </tr>
<?php
            } catch (PDOException $e) {
    die ("Error!: " . $e->getMessage() . "<br/>");
}
            ?>
```

运行结果如图 21.4 所示。

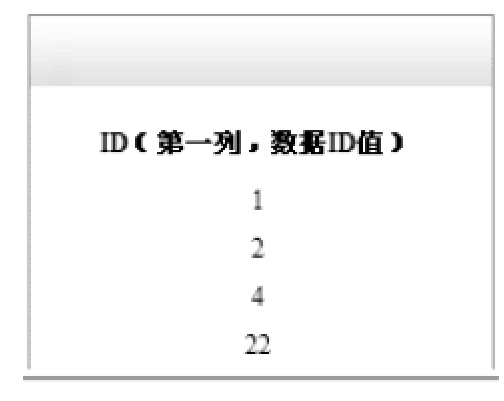


图 21.4　fetchColumn()方法获取结果集中第一列的值

21.5 PDO 中捕获 SQL 语句中的错误

视频讲解：光盘\TM\Video\第 21 章\PDO 中捕获 SQL 语句中的错误.exe

在 PDO 中捕获 SQL 语句中的错误有 3 种方案可以选择。

21.5.1 使用默认模式——PDO::ERRMODE_SILENT

在默认模式中设置 PDOStatement 对象的 errorCode 属性，但不进行其他任何操作。

例 21.4 通过 prepare()和 execute()方法向数据库中添加数据，设置 PDOStatement 对象的 errorCode 属性，手动检测代码中的错误。**（实例位置：光盘\TM\Instance\21\21.4）**

创建 index.php 文件，添加 form 表单，将表单元素提交到本页。通过 PDO 连接 MySQL 数据库，通过预处理语句 prepare()和 execute()执行 INSERT 添加语句，向数据表中添加数据，并且设置 PDOStatement 对象的 errorCode 属性，检测代码中的错误。其关键代码如下。

```
<?php
if($_POST['Submit']=="提交" && $_POST['pdo']!=""){
	$dbms='mysql';	//数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
	$host='localhost';				//数据库主机名
	$dbName='db_database21';			//使用的数据库
	$user='root';					//数据库连接用户名
	$pass='111';					//对应的密码
	$dsn="$dbms:host=$host;dbname=$dbName";
	$pdo = new PDO($dsn, $user, $pass);		//初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
	$query="insert into
tb_pdo_mysqls(pdo_type,database_name,dates)values('".$_POST['pdo']."','".$_POST['databases']."','".$_POST
['dates']."')";
	$result=$pdo->prepare($query);
	$result->execute();
	$code=$result->errorCode();
	if(empty($code)){
		echo "数据添加成功！";
	}else{
		echo '数据库错误：<br/>';
		echo 'SQL Query:'.$query;
		echo '<pre>';
		var_dump($result->errorInfo());
		echo '</pre>';
	}
}
?>
```

在本实例中，在定义 INSERT 添加语句时，使用了错误的数据表名称 tb_pdo_mysqls（正确名称是 tb_pdo_mysql），导致输出结果如图 21.5 所示。

21.5.2 使用警告模式——PDO::ERRMODE_WARNING

警告模式会产生一个 PHP 警告，并设置 errorCode 属性。如果设置的是警告模式，那么除非明确地检查



错误代码，否则程序将继续按照其方式运行。

例 21.5　设置警告模式，通过 prepare()和 execute()方法读取数据库中数据，并且通过 while 语句和 fetch()方法完成数据的循环输出，体会在设置成警告模式后执行错误的 SQL 语句。（**实例位置：光盘\TM\Instance\21\21.5**）

创建 index.php 文件，连接 MySQL 数据库，通过预处理语句 prepare()和 execute()执行 SELECT 查询语句，并设置一个错误的数据表名称，同时通过 setAttribute()方法设置为警告模式，最后通过 while 语句和 fetch()方法完成数据的循环输出。其关键代码如下：

```
<?php
$dbms='mysql';     //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
$host='localhost';                                   //数据库主机名
$dbName='db_database21';                             //使用的数据库
$user='root';                                        //数据库连接用户名
$pass='111';                                         //对应的密码
$dsn="$dbms:host=$host;dbname=$dbName";
try {
    $pdo = new PDO($dsn, $user, $pass);              //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
     $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_WARNING);          //设置为警告模式
     $query="select * from tb_pdo_mysqls";           //定义 SQL 语句
     $result=$pdo->prepare($query);                  //准备查询语句
     $result->execute();                             //执行查询语句，并返回结果集
     while($res=$result->fetch(PDO::FETCH_ASSOC)){ //while 循环输出查询结果集，并且设置结果集的为关联索引
     ?>
          <tr>
            <td height="22" align="center" valign="middle"><?php echo $res['id'];?></td>
            <td align="center" valign="middle"><?php echo $res['pdo_type'];?></td>
            <td align="center" valign="middle"><?php echo $res['database_name'];?></td>
            <td align="center" valign="middle"><?php echo $res['dates'];?></td>
          </tr>
<?php
     }
            } catch (PDOException $e) {
    die ("Error!: " . $e->getMessage() . "<br/>");
}
            ?>
```

在设置为警告模式后，如果 SQL 语句出现错误将给出一个提示信息，但是程序仍能够继续执行下去，其运行结果如图 21.6 所示。

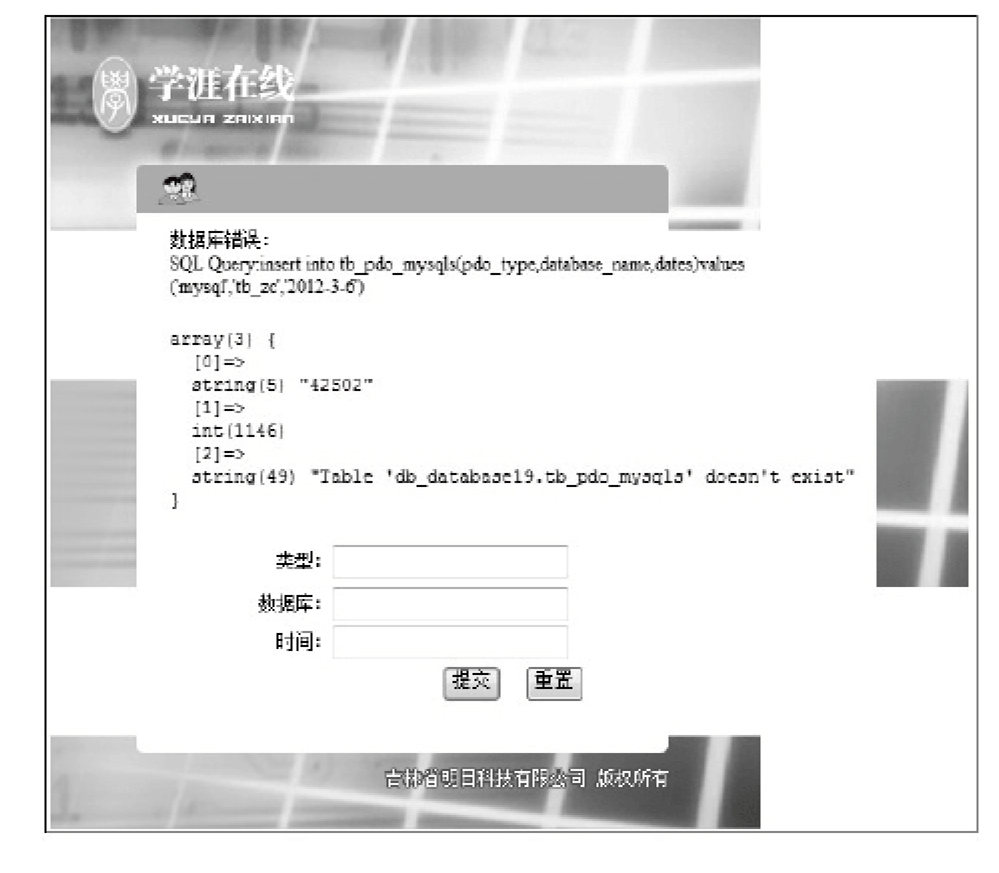


图 21.5　在默认模式中捕获 SQL 中的错误

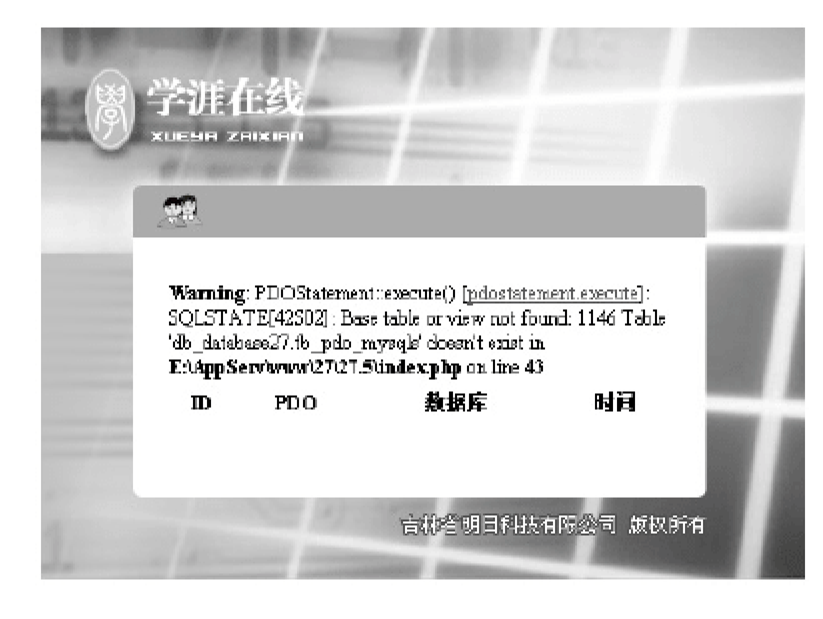


图 21.6　设置警告模式后捕获的 SQL 语句错误

21.5.3 使用异常模式——PDO::ERRMODE_EXCEPTION

异常模式会创建一个 PDOException，并设置 errorCode 属性。它可以将执行代码封装到一个 try…catch 语句块中。未捕获的异常将会导致脚本中断，并显示堆栈跟踪让你了解是哪里出现的问题。

例 21.6 在执行数据库中数据的删除操作时，设置为异常模式，并且编写一个错误的 SQL 语句（操作错误的数据表 tb_pdo_mysqls），体会异常模式与警告模式和默认模式的区别。（**实例位置：光盘\TM\Instance\21\21.6**）

（1）创建 index.php 文件，连接 MySQL 数据库，通过预处理语句 prepare()和 execute()执行 SELECT 查询语句，再通过 while 语句和 fetch()方法完成数据的循环输出，并且设置删除超链接，链接到 delete.php 文件，传递的参数是数据的 ID 值。其运行效果如图 21.7 所示。



图 21.7 数据的循环输出

（2）创建 delete.php 文件，获取超链接传递的数据 ID 值。连接数据库，通过 setAttribute()方法设置为异常模式，定义 DELETE 删除语句，删除一个错误数据表（tb_pdo_mysqls）中的数据，并且通过 try…catch 语句捕获错误信息。其代码如下：

```
<?php
header ( "Content-type: text/html; charset=utf-8" );    //设置文件编码格式
if($_GET['conn_id']!=""){
    $dbms='mysql';   //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
    $host='localhost';                              //数据库主机名
    $dbName='db_database21';                        //使用的数据库
    $user='root';                                   //数据库连接用户名
    $pass='111';                                    //对应的密码
    $dsn="$dbms:host=$host;dbname=$dbName";
    try {
    $pdo = new PDO($dsn, $user, $pass);             //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
        $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION);
        $query="delete from tb_pdo_mysqls where Id=:id";
        $result=$pdo->prepare($query);                  //预准备语句
        $result->bindParam(':id',$_GET['conn_id']);     //绑定更新的数据
        $result->execute();
    } catch (PDOException $e) {
        echo 'PDO Exception Caught.';
        echo 'Error with the database:<br/>';
        echo   'SQL Query: '.$query;
        echo '<pre>';
    echo "Error: " . $e->getMessage(). "<br/>";
        echo "Code: " . $e->getCode(). "<br/>";
        echo "File: " . $e->getFile(). "<br/>";
        echo "Line: " . $e->getLine(). "<br/>";
        echo "Trace: " . $e->getTraceAsString(). "<br/>";
        echo '</pre>';
    }
}
?>
```

在设置为异常模式后，执行错误的 SQL 语句返回的结果如图 21.8 所示。

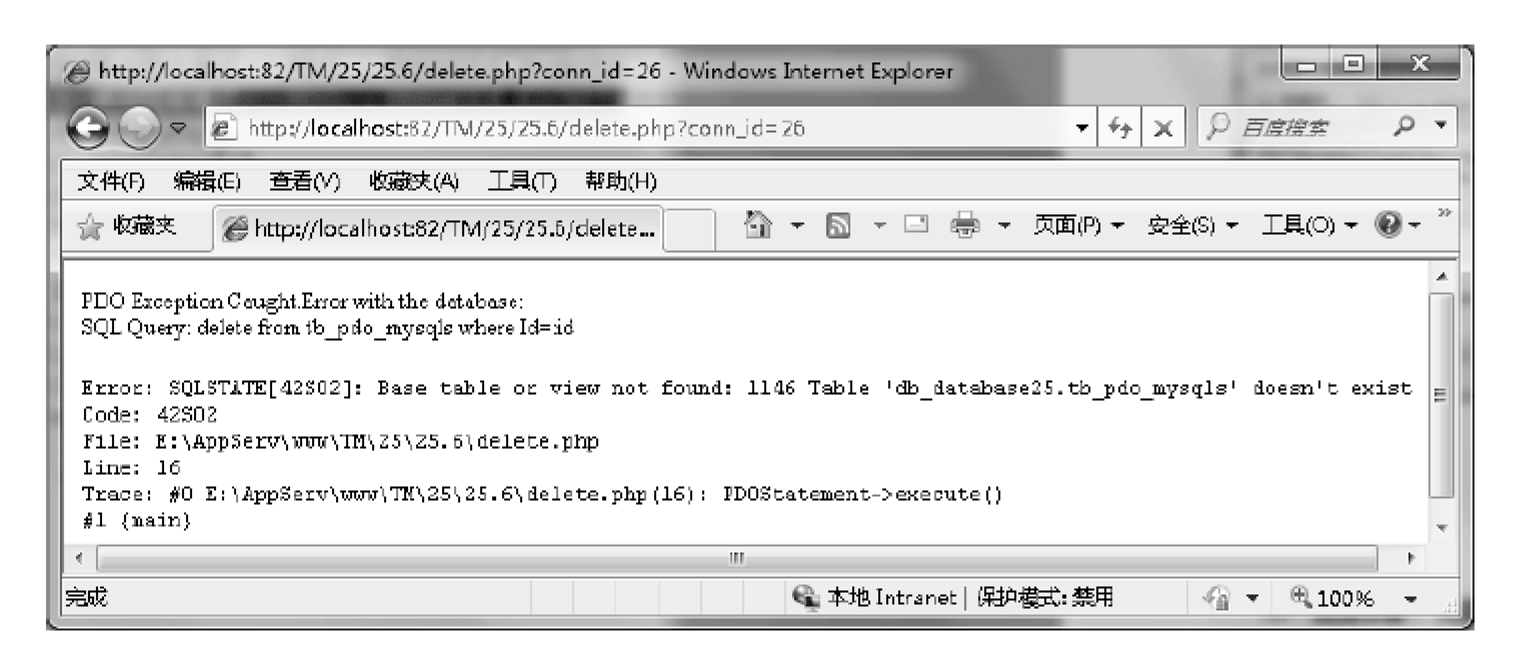


图 21.8　异常模式捕获的 SQL 语句错误信息

21.6　PDO 中的错误处理

视频讲解：光盘\TM\Video\第 21 章\PDO 中错误处理.exe

在 PDO 中有两个获取程序中错误信息的方法：errorCode()方法和 errorInfo()方法。

21.6.1　errorCode()方法

errorCode()方法用于获取在操作数据库句柄时所发生的错误代码，这些错误代码被称为 SQLSTATE 代码。其语法格式如下：

```
int PDOStatement::errorCode ( void )
```

errorCode()方法返回一个 SQLSTATE，SQLSTATE 是由 5 个数字和字母组成的代码。

例 21.7　在 PDO 中通过 query()方法完成数据的查询操作，并且通过 foreach 语句完成数据的循环输出。在定义 SQL 语句时使用一个错误的数据表，并且通过 errorCode()方法返回错误代码。（**实例位置：光盘\TM\Instance\21\21.7**）

创建 index.php 文件。首先通过 PDO 连接 MySQL 数据，然后通过 query()方法执行查询语句，接着通过 errorCode()方法获取错误代码，最后通过 foreach 语句完成数据的循环输出。其关键代码如下。

```
<?php
$dbms='mysql';      //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
$host='localhost';                              //数据库主机名
$dbName='db_database21';                        //使用的数据库
$user='root';                                   //数据库连接用户名
$pass='111';                                    //对应的密码
$dsn="$dbms:host=$host;dbname=$dbName";
try {
$pdo = new PDO($dsn, $user, $pass);             //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
$query="select * from tb_pdo_mysqls";           //定义 SQL 语句
$result=$pdo->query($query);                    //执行查询语句，并返回结果集
echo "errorCode 为：".$pdo->errorCode();
foreach($result as $items){
     ?>
          <tr>
            <td height="22" align="center" valign="middle"><?php echo $items['id'];?></td>
            <td align="center" valign="middle"><?php echo $items['pdo_type'];?></td>
```

```
                <td align="center" valign="middle"><?php echo $items['database_name'];?></td>
                <td align="center" valign="middle"><?php echo $items['dates'];?></td>
              </tr>
               <?php
               }
        } catch (PDOException $e) {
       die ("Error!: " . $e->getMessage() . "<br/>");
}
 ?>
```

运行结果如图 21.9 所示。

21.6.2　errorInfo()方法

errorInfo()方法用于获取操作数据库句柄时所发生的错误信息。其语法格式如下：

```
array PDOStatement::errorInfo ( void )
```

errorInfo()方法的返回值为一个数组，它包含了相关的错误信息。

例 21.8　在 PDO 中通过 query()方法完成数据的查询操作，并且通过 foreach 语句完成数据的循环输出。在定义 SQL 语句时使用一个错误的数据表，并且通过 errorInfo()方法返回错误信息。**（实例位置：光盘\TM\Instance\21\21.8）**

创建 index.php 文件。首先通过 PDO 连接 MySQL 数据库，然后通过 query()方法执行查询语句，接着通过 errorInfo()方法获取错误信息，最后通过 foreach 语句完成数据的循环输出。其关键代码如下：

```
<?php
$dbms='mysql';      //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
$host='localhost';                                  //数据库主机名
$dbName='db_database21';                            //使用的数据库
$user='root';                                       //数据库连接用户名
$pass='111';                                        //对应的密码
$dsn="$dbms:host=$host;dbname=$dbName";
try {
$pdo = new PDO($dsn, $user, $pass);                 //初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
           $query="select * from tb_pdo_mysqls";    //定义 SQL 语句
           $result=$pdo->query($query);             //执行查询语句，并返回结果集
           print_r($pdo->errorInfo());
      foreach($result as $items){
      ?>
             <tr>
                <td height="22" align="center" valign="middle"><?php echo $items['id'];?></td>
                <td align="center" valign="middle"><?php echo $items['pdo_type'];?></td>
                <td align="center" valign="middle"><?php echo $items['database_name'];?></td>
                <td align="center" valign="middle"><?php echo $items['dates'];?></td>
             </tr>
               <?php
               }
               } catch (PDOException $e) {
     die ("Error!: " . $e->getMessage() . "<br/>");
}
?>
```

运行结果如图 21.10 所示。



图 21.9　通过 errorCode()方法获取错误代码



图 21.10　通过 errorInfo()方法获取错误信息

21.7　PDO 中的事务处理

视频讲解：光盘\TM\Video\第 21 章\PDO 中事务处理.exe

在 PDO 中同样可以实现事务处理的功能，其应用的方法如下：

☑　开启事务—— beginTransaction()方法

beginTransaction()方法将关闭自动提交（autocommit）模式，直到事务提交或者回滚以后才恢复。

☑　提交事务—— commit()方法

commit()方法完成事务的提交操作，成功则返回 TRUE，否则返回 FALSE。

☑　事务回滚—— rollback()方法

rollback()方法执行事务的回滚操作。

例 21.9　通过 prepare()和 execute()方法向数据库中添加数据，并且通过事务处理机制确保数据能够正确地添加到数据中。**（实例位置：光盘\TM\Instance\21\21.9）**

创建 index.php 文件。首先，定义数据库连接的参数，创建 try…catch 语句，在 try 语句中实例化 PDO 构造函数，完成与数据库的连接，并且通过 beginTransaction()方法开启事务。然后，定义 INSERT 添加语句，通过$_POST[]方法获取表单中提交的数据，通过 prepare()和 execute()方法向数据库中添加数据，并且通过 commit()方法完成事务的提交操作。最后，在 catch 语句中返回错误信息，并且通过 rollBack()执行事务的回滚操作。其代码如下：

```
<?php
if($_POST['Submit']=="提交" && $_POST['pdo']!=""){
	$dbms='mysql';	//数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
	$host='localhost';				//数据库主机名
	$dbName='db_database21';			//使用的数据库
	$user='root';					//数据库连接用户名
	$pass='111';					//对应的密码
	$dsn="$dbms:host=$host;dbname=$dbName";
	try {
	$pdo = new PDO($dsn, $user, $pass);		//初始化一个 PDO 对象，就是创建了数据库连接对象$pdo
		$pdo->beginTransaction();		//开启事务
		$query="inser into tb_pdo_mysql(pdo_type,database_name,dates)values('".$_POST['pdo']."','".$_POST
['databases']."','".$_POST['dates']."')";
		$result=$pdo->prepare($query);
		if($result->execute()){
			echo "数据添加成功！";
		}else{
```

```
                echo "数据添加失败！";
            }
            $pdo->commit();                          //执行事务的提交操作
        } catch (PDOException $e) {
            die ("Error!: " . $e->getMessage() . "<br/>");
        $pdo->rollBack();                            //执行事务的回滚
        }
}
?>
```

运行结果如图 21.11 所示。



图 21.11　数据添加中应用事务处理机制

21.8　PDO 中的存储过程

视频讲解：光盘\TM\Video\第 21 章\PDO 中存储过程.exe

存储过程允许在更接近于数据的位置操作数据，从而减少带宽的使用，它们使数据独立于脚本逻辑，允许使用不同语言的多个系统以相同的方式访问数据，从而节省了花费在编码和调试上的宝贵时间。同时它使用预定义的方案执行操作，提高查询速度，并且能够阻止与数据的直接相互作用，从而起到保护数据的作用。

下面讲解如何在 PDO 中调用存储过程。这里首先创建一个存储过程，其 SQL 语句如下：

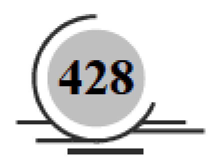
```
drop procedure if exists pro_reg;
delimiter //
create procedure pro_reg (in nc varchar(80), in pwd varchar(80), in email varchar(80),in address varchar(50))
begin
insert into tb_reg (name, pwd ,email ,address) values (nc, pwd, email, address);
end;
//
```

☑　drop 语句删除 MySQL 服务器中已经存在的存储过程 pro_reg。
☑　delimiter //的作用是将语句结束符更改为“//”。
☑　in nc varchar(50)…in address varchar(50)表示要向存储过程中传入的参数。
☑　begin…end 表示存储过程中的语句块，它的作用类似与 PHP 语言中的“{…}”。

存储过程创建成功后，下面调用这个存储过程实现用户注册的功能。

例 21.10　在 PDO 中通过 CALL 语句调用存储过程，实现用户注册信息的添加操作。（**实例位置：光盘\TM\Instance\21\21.10**）

创建 index.php 文件。首先，创建 form 表单，将用户注册信息通过 POST 方法提交到本页。然后，在本



页中编写 PHP 脚本，通过 PDO 连接 MySQL 数据库，并且设置数据库编码格式为 utf-8，获取表单中提交的用户注册信息。接着，通过 call 语句调用存储过程 pro_reg，将用户注册信息添加到数据表中。最后，通过 try…catch 语句块返回错误信息。其关键代码如下：

```
<?php
 if($_POST['submit']!=""){
 $dbms='mysql';      //数据库类型，对于开发者来说，使用不同的数据库，只要改这个，不用记住那么多的函数
     $host='localhost';                                     //数据库主机名
     $dbName='db_database21';                               //使用的数据库
     $user='root';                                          //数据库连接用户名
     $pass='111';                                           //对应的密码
     $dsn="$dbms:host=$host;dbname=$dbName";
     try {
     $pdo = new PDO($dsn, $user, $pass);                    //初始化一个 PDO 对象，就是创建了数据库连接对象
$pdo
          $pdo->query("set names utf8");                    //设置数据库编码格式
          $pdo->setAttribute(PDO::ATTR_ERRMODE,PDO::ERRMODE_EXCEPTION); //定义错误异常模式
     $nc=$_POST['nc'];
     $pwd=md5($_POST['pwd']);
     $email=$_POST['email'];
     $address=$_POST['address'];
          $query="call pro_reg('$nc','$pwd','$email','$address')";
          $result=$pdo->prepare($query);
          if($result->execute()){
               echo "数据添加成功！ ";
          }else{
               echo "数据添加失败！ ";
          }
     } catch (PDOException $e) {
     echo 'PDO Exception Caught.';
          echo 'Error with the database:<br/>';
          echo   'SQL Query: '.$query;
          echo '<pre>';
     echo "Error: " . $e->getMessage(). "<br/>";
          echo "Code: " . $e->getCode(). "<br/>";
          echo "File: " . $e->getFile(). "<br/>";
          echo "Line: " . $e->getLine(). "<br/>";
          echo "Trace: " . $e->getTraceAsString(). "<br/>";
          echo '</pre>';
     }
 }
?>
```

其运行结果如图 21.12 所示。

图 21.12　通过存储过程完成用户的注册

21.9 实 战

视频讲解：光盘\TM\Video\第 21 章\实战.exe

21.9.1 通过 PDO 更新数据库中数据

例 21.11 在利用 MySQL 数据库函数更新数据时，需要将要更改的数据信息 id 拼接到地址栏的参数中。在 PDO 中操作数据更新也不例外。虽然利用 PDO 操作数据的方法是不同的，但是原理是相同的。（**实例位置：光盘\TM\Instance\21\21.11**）

首先，创建脚本文件 index.php，编写表格，设置 PDO 抽象层的相关参数拼接$dsn 变量。其次，判断“修改”按钮是否被单击，当“修改”按钮被单击时，获取地址栏传递的参数并在表格最下方动态创建 form 表单，其中包括 3 个文本框和一个提交按钮。文本框中的 value 属性值由 PHP 提供数据并设置 id 的文本框，将其设置为只读属性不可更改。最后，当单击“确定”按钮时，通过 PDO 的 exec()方法执行更新操作。其代码如下：

```
<?php
	try {
		$pdo=new PDO($dsn,$user,$pwd);                          //实例化对象
		$sql="select * from tb_pdo";                            //查询 SQL 语句
		$result=$pdo->query($sql);                              //返回结果集
		foreach($result as $value){
?>
	<tr>
		<td class="one"><?php echo $value[0];?></td>            //循环输出数据
		<td class="one"><?php echo $value[1];?></td>
		<td class="one"><?php echo $value[2];?></td>
		<td class="one"><?php echo $value[3];?></td>
		<td class="one"><a href="index.php?id=<?php echo $value[0];?>">修改</a></td>    //拼接地址栏参数
	</tr>
<form action="method="post">
<?php
		}
		if(isset($_GET[id])){                                   //当页面中存在 id 变量时
			$sql1="select * from tb_pdo where id='$_GET[id]'";  //拼接 SQL 语句
			$resul=$pdo->query($sql1);                          //返回结果集
				foreach ($resul as $value){                     //循环输出结果


?>
	<tr>
			<td class="one"><input readonly type='text' name='id' value='<?php echo $value[id];?>'></td>
			<td class="one"><input type='text' name='user' value='<?php echo $value[1];?>'></td>
			<td class="one"><input type='password' name='pwd' value='<?php echo $value[2];?>'></td>
			<td class="one"><input type='text' name='date' value='<?php echo $value[3];?>'></td>
			<td class="one"><input class="two" type='submit' name='sub1' value='确定'></td>
	</tr>

<?php
```

```
                }
            }
?>
    </form>
<?php
        if($_POST[sub1]=="确定"){                                    //当页面中存在参数 sub1 时
                $sql2="update tb_pdo set username='$_POST[user]',userpwd='$_POST[pwd]',date='$_POST
[date]' where id='$_POST[id]'";                                     //拼接 SQL 语句
                $resu=$pdo->exec($sql2);                            //执行更新操作
                if($resu==1){
                    echo "<script>alert('更新数据成功');window.location.href='index.php'</script>";
                }
            }
    } catch (Exception $e) {
        echo "ERROR!!!".$e->getMessage()."<br>";
    }
?>
```

其运行结果如图 21.13 和图 21.14 所示。

ID	用户名称	用户密码	操作时间	修改
1	mr	mrsoft	2011-04-09	修改
2	mrsoft	mrkj	2011-11-24	修改
3	yu	666666	2010-11-24	修改
3	mingribook	••••••	2012-11-24	确定

图 21.13　更新数据



图 21.14　更新数据成功

21.9.2　明日书店会员注册

例 21.12　注册系统在本书的很多节中都有体现，本次所讲解的是利用 PDO 抽象层知识实现的会员注册系统。其核心思想是拼接插入的 SQL 代码，将文本框的相关信息动态的添加到数据表中。（**实例位置：光盘\TM\Instance\21\21.12**）

首先，创建脚本文件 index.php。在脚本文件中编写表单，设置一个“注册”按钮和一个“重置”按钮。其次，当单击“注册”按钮时，首先通过 PDO 连接 MySQL 数据库，实例化 PDO 对象并利用对象句柄调用 exec()方法，向数据库中添加数据。其代码如下：

```
<?php
    $dbms='mysql';                                          //定义 PDO 的相关参数
    $host='localhost';
    $user='root';
    $pwd='111';
    $dbName='db_database21';
    $dsn="$dbms:host=$host;dbname=$dbName";
    if(isset($_POST[sub])){                                  //当页面中存在 sub 变量时
        try {
            $pdo=new PDO($dsn,$user,$pwd);                  //实例化对象
            $sql="insert into tb_zc values('','$_POST[text]','$_POST[pwd]','$_POST[qq]','$_POST[mail]',now())";
                                                            //拼接 SQL 语句
            $result=$pdo->exec($sql);                       //返回结果集
            if($result==1){                                 //输出提示
                echo "<script>alert('注册成功');</script>";
```

```
            }
        } catch (Exception $e) {
            echo "ERROR".$e->getMessage()."<br>";              //输出异常
        }
    }
?>
```

运行效果如图 21.15 所示。

21.9.3　添加留言信息

例 21.13　添加留言信息是非常普通和常见的模块操作，其实从原理上说与注册登录操作几乎一样，笔者意在告知读者 PDO 操作常见的几种模式，使用读者熟能生巧。**（实例位置：光盘\TM\Instance\21\21.13）**

本应用与注册登录模块原理相同，这里就不在阐述操作步骤了。本应用的核心代码如下：

```
<?php
    $dbms='mysql';                                        //定义 PDO 的相关参数
    $user='root';
    $pwd='111';
    $host='localhost';
    $dbName='db_database21';
    $dsn="$dbms:host=$host;dbname=$dbName";
    if(isset($_POST[sub])){                               //当页面中存在 sub 变量时
        try {                                             //利用 try…catch 捕获异常
            $pdo = new PDO($dsn,$user,$pwd);              //实例化对象
            $pdo->query("SET NAMES utf8");                //设置页面的编码集风格
            $sql="insert into tb_fb values('','$_POST[title]','$_POST[content]',now())";  //定义 SQL 语句
            $rs=$pdo->exec($sql);                                                      //执行插入操作
            if($rs=="1"){
                echo "<b>新闻发布成功</b>";
            }
        } catch (Exception $e) {
            echo "ERROR".$e->getMessage()."<br>";
        }
    }
?>
```

运行效果如图 21.16 所示。



图 21.15　会员注册系统

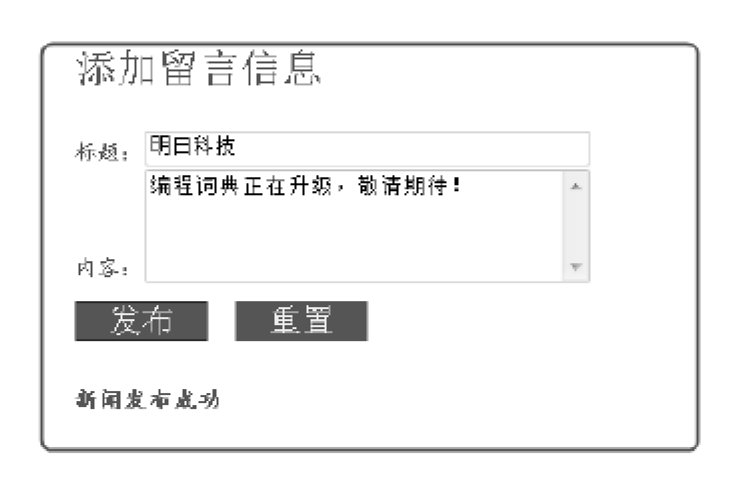


图 21.16　添加留言信息



21.9.4　查询留言内容

例 21.14　本应用中的查询留言内容模块是一个典型的站内搜索关键字描红的基础操作。究其根源，其实 PDO 的操作并不复杂，它只是提供了一个连接多种数据库的一个统一的接口。至于具体的操作还是来自于 SQL 语句本身。**（实例位置：光盘\TM\Instance\21\21.14）**

操作步骤如下。

（1）创建脚本文件 index.php。定义 form 表单，设置一个文本框和一个“查询”按钮。

（2）当“查询”按钮被单击时，首先，判断文本框内容是否为空，其次，使用 PDO 抽象层连接 MySQL 数据库，并在 try…catch 内部利用 PDO 对象句柄调用 query()函数执行查询操作，最后，输出查询结果。其核心代码如下：

```
<?php
	if(isset($_POST[sub])){                                    //判断页面是否存在 sub 变量
		if($_POST[text]=="" || $_POST[text]=="输入查询内容"){     //判断文本框内容是否为空
			echo "文本框内容不能为空";
		}else{
			$dbms="mysql";                                       //定义 PDO 的相关参数
			$user="root";
			$pwd="111";
			$host="localhost";
			$dbName="db_database21";
			$dsn="$dbms:host=$host;dbname=$dbName";
			try {                                                          //try…catch 捕获异常
				$pdo = new PDO($dsn,$user,$pwd);                           //实例化对象
				$sql="select * from tb_fb where title like '%".$_POST['text']."%'";   //拼接 SQL 语句
				$pdo->query("SET NAMES utf8");                   //设置页面数据的编码风格
				$rs=$pdo->query($sql);
				foreach($rs as $value){                          //将数据循环输出
	?>
		<tr>
			<td class="f"><?php echo $rs [0];?></td>
			<td class="f"><?php echo $rs [1];?></td>
			<td class="f"><?php echo $rs [2];?></td>
			<td class="f"><?php echo $rs [3];?></td>
		</tr>
<?php
				}
			} catch (Exception $e) {
				echo "ERROR".$e->getMessage()."<br>";
			}
		}
	}
?>
```

运行效果如图 21.17 所示。



图 21.17　查询留言内容

21.10 小　　结

本章重点介绍数据库抽象层——PDO，从它的概述、特点和安装开始讲解，到它的实际应用，包括如何连接不同的数据库、如何执行 SQL 语句、如何获取结果集，以及错误处理，再到它的高级应用事务和存储过程都进行了详细讲解，并且都配有相应的实例。通过本章的学习，相信读者能够掌握 PDO 技术的应用。

21.11 学习成果检验

1．通过 PDO 向数据库中添加数据。**（实例位置：光盘\TM\Instance\21\21.15）**

2．通过 PDO 浏览数据库中数据。**（答案位置：光盘\TM\Instance\21\21.16）**

第22章

综合实例（四）——BCTY365网上社区

（视频讲解：138分钟）

所谓网上社区是指包括BBS/论坛、聊天室、博客等形式在内的网上交流空间，同一主题的网上社区集中了具有相同兴趣的访问者，由于有众多用户的参与，因此具备交流的功能。

网上社区有各种不同的表现形式和规模，由个人创办的社区，功能和界面追求时尚、个性突出；大型的商业性质社区，以盈利为目的，分类多元化，适合不同类型的网民。

通过阅读本章内容，你可以：

- 了解网上社区开发的基本过程
- 了解如何做需求分析和系统设计
- 了解如何设计和实现下载功能
- 了解在线论坛功能的实现方法
- 了解注册模块设计
- 了解技术支持模块设计
- 了解后台首页设计
- 了解在线支付技术

22.1 BCTY365 网上社区概述

本章开发的 BCTY365 网上社区主要面向程序开发人员，集论坛、留言板、软件下载、升级下载、技术支持和在线购物等功能于一身，既是一个程序开发者交流的平台，也是一个网络营销的场所。

22.1.1 系统功能结构流程

BCTY365 网上社区系统主要分为前台和后台两部分，为了使读者能够更清楚地了解网站的结构，下面分别给出 BCTY365 网上社区前台和后台功能模块结构图。

BCTY365 网上社区前台管理系统的功能设计如图 22.1 所示。

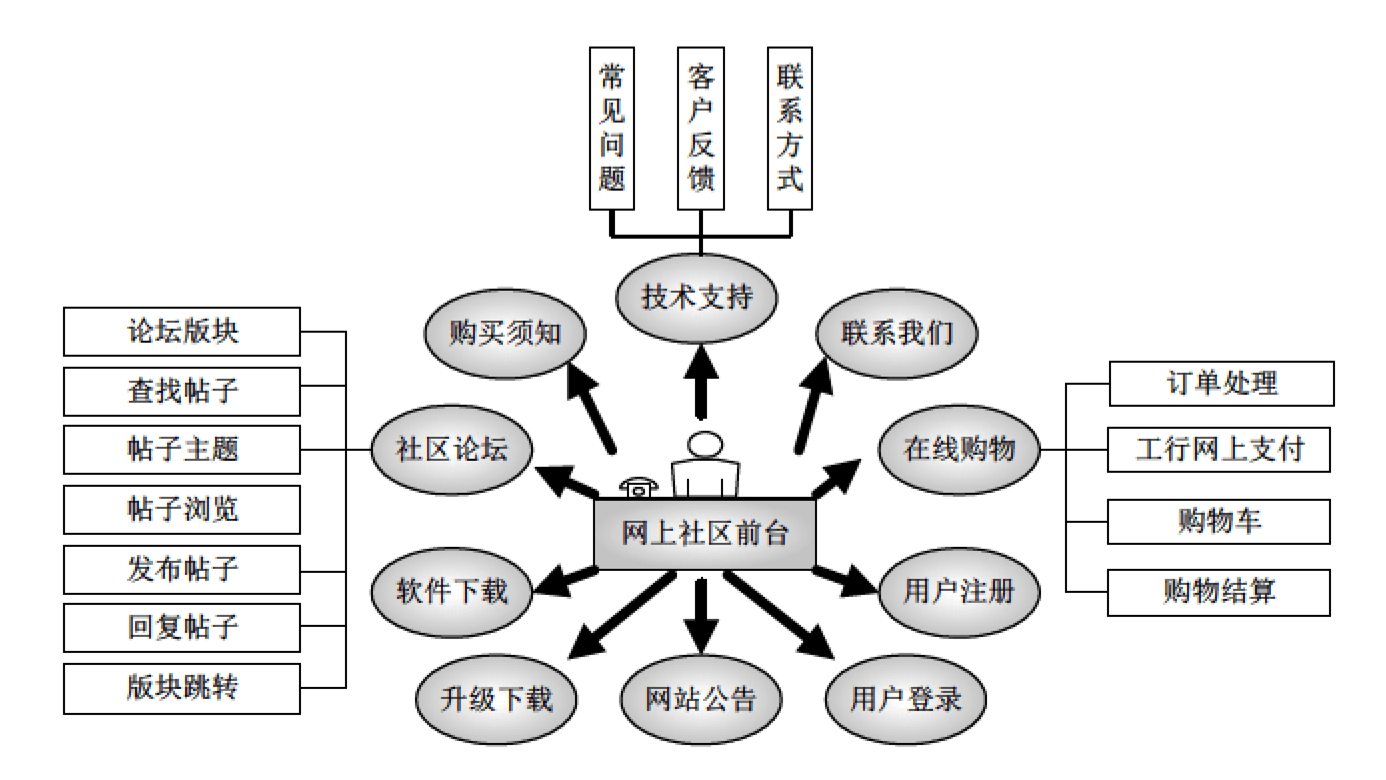


图 22.1 网上社区前台功能模块结构图

BCTY365 网上社区后台管理系统的功能设计如图 22.2 所示。

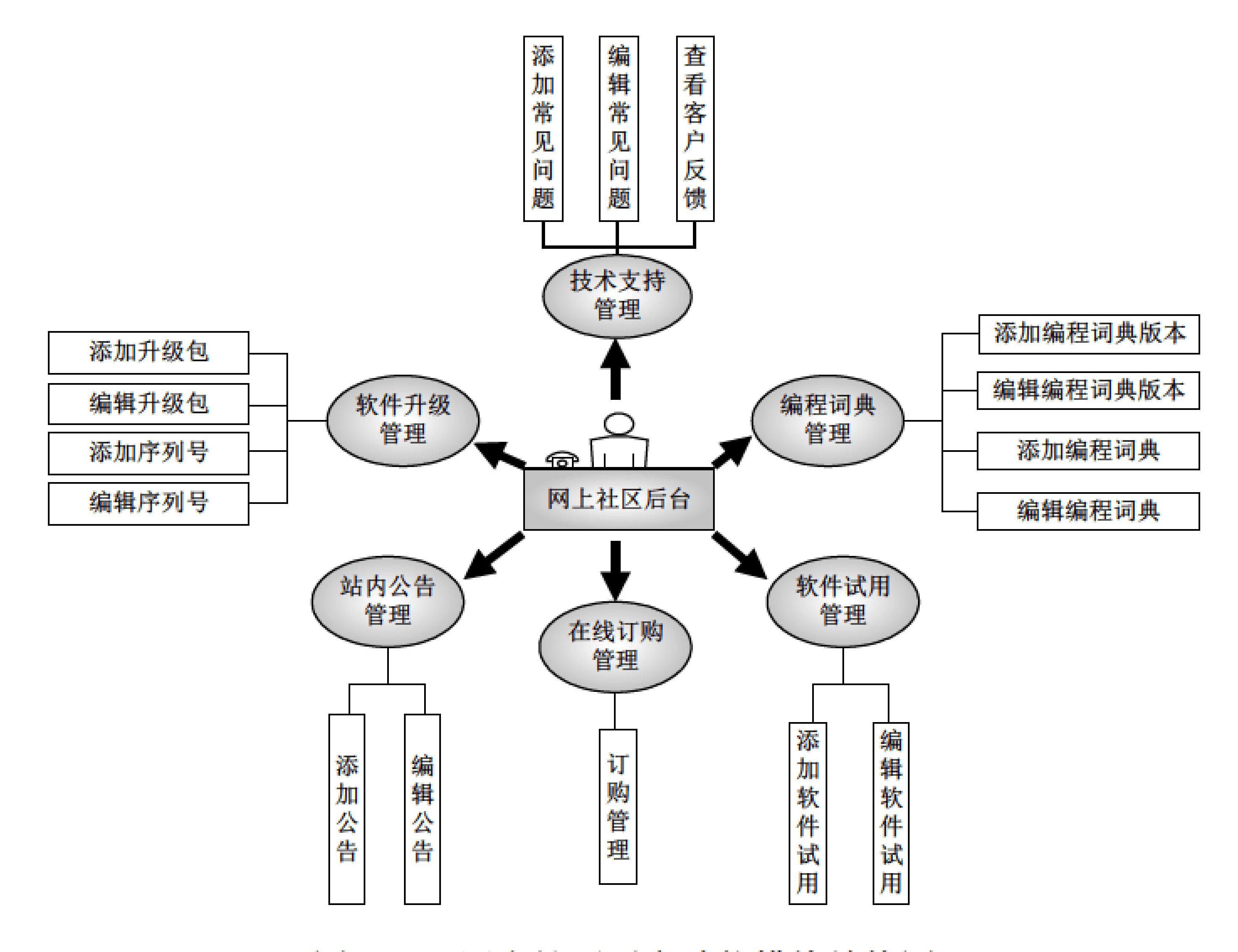


图 22.2 网上社区后台功能模块结构图

22.1.2　系统预览

BCTY365 网上社区系统由多个程序页面组成，下面给出几个典型页面，其他页面参见光盘中的源程序。

前台首页如图 22.3 所示，该页面用于展示本系统的功能模块，突出企业的形象，推广企业的软件产品。后台首页如图 22.4 所示，该页面用于实现对编程词典、技术支持、软件升级、软件试用等内容的管理。



图 22.3　前台首页



图 22.4　后台首页

在线订购模块的页面效果如图 22.5 所示，该页面主要用于展示本企业在线推出的软件产品，实现对产品的在线购买功能。软件下载模块的页面效果如图 22.6 所示，该页面主要用于展示本企业提供的免费软件，并且提供下载链接。



图 22.5　在线订购



图 22.6　软件下载

社区论坛模块的页面效果如图 22.7 所示，该页面主要用于展示论坛中的各大版块，并且提供超链接跳转到对应的版块。后台的登录页面效果如图 22.8 所示，该页面主要实现后台管理员的登录。



图 22.7　社区论坛



图 22.8　后台登录

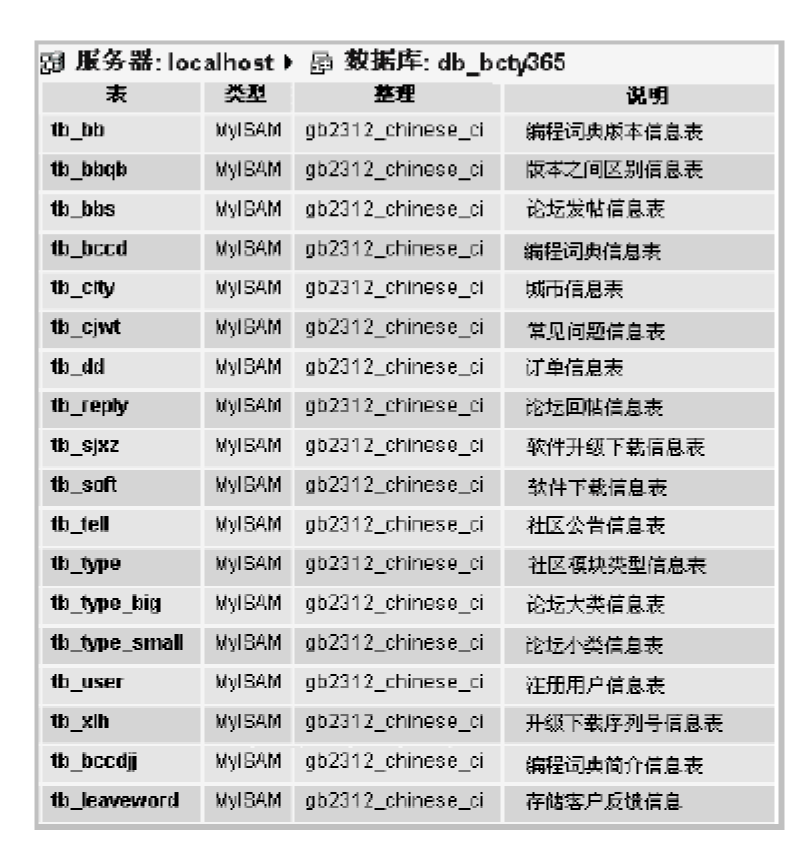
22.2　数据库设计

视频讲解：光盘\TM\Video\第 22 章\数据库设计.exe

开发一个功能完善的网上社区离不开数据库的支持，只有拥有了强大的数据库，网上社区才能够存储大量的数据信息，实现更多、更好的功能来吸引更多的社区成员。本节将对 BCTY365 网上社区数据库的设计进行详细介绍。

22.2.1　数据库概要说明

在 BCTY365 网上社区系统中应用的是 db_bcty365 数据库，其中涉及 18 个数据表，数据表的名称和功能如图 22.9 所示。



服务器: localhost ▸ 数据库: db_bcty365

表	类型	整理	说明
tb_bb	MyISAM	gb2312_chinese_ci	编程词典版本信息表
tb_bbqb	MyISAM	gb2312_chinese_ci	版本之间区别信息表
tb_bbs	MyISAM	gb2312_chinese_ci	论坛发帖信息表
tb_bccd	MyISAM	gb2312_chinese_ci	编程词典信息表
tb_city	MyISAM	gb2312_chinese_ci	城市信息表
tb_cjwt	MyISAM	gb2312_chinese_ci	常见问题信息表
tb_dd	MyISAM	gb2312_chinese_ci	订单信息表
tb_reply	MyISAM	gb2312_chinese_ci	论坛回帖信息表
tb_sjxz	MyISAM	gb2312_chinese_ci	软件升级下载信息表
tb_soft	MyISAM	gb2312_chinese_ci	软件下载信息表
tb_tell	MyISAM	gb2312_chinese_ci	社区公告信息表
tb_type	MyISAM	gb2312_chinese_ci	社区模块类型信息表
tb_type_big	MyISAM	gb2312_chinese_ci	论坛大类信息表
tb_type_small	MyISAM	gb2312_chinese_ci	论坛小类信息表
tb_user	MyISAM	gb2312_chinese_ci	注册用户信息表
tb_xlh	MyISAM	gb2312_chinese_ci	升级下载序列号信息表
tb_bccdjj	MyISAM	gb2312_chinese_ci	编程词典简介信息表
tb_leaveword	MyISAM	gb2312_chinese_ci	存储客户反馈信息

图 22.9　db_bcty365 数据库中使用的数据表

22.2.2　数据库概念设计

根据上述各节对 BCTY365 网上社区系统做的需求分析和系统设计，规划出 BCTY365 网上社区的实体关系 E-R 图。其中包括注册用户实体、发帖信息实体、回帖信息实体、订单信息实体、编程词典信息实体，以及一些辅助实体，用于对上述实体进行补充。由于涉及的实体较多，这里只对注册用户实体、发帖信息实体和订单信息实体的 E-R 图进行介绍。

1．注册用户实体

注册用户实体用于存储用户注册信息，包括编号、用户名、真实姓名、密码、邮箱、性别、电话、QQ 号码、家庭地址、访问次数、注册时间、最后一次登录时间、IP 地址、邮政编码、用户类型、密码提示问题、密码答案、真实密码、表情图和发帖次数属性。注册用户实体的 E-R 图如图 22.10 所示。

2．发帖信息实体

发帖信息实体用于存储登录本社区的会员在论坛中发布帖子的相关信息，包括编号、用户名 ID、帖子类型、帖子标题、帖子内容、发帖时间、最后回复时间、表情图、访问次数、是否顶帖和上传图片属性。发帖信息实体的 E-R 图如图 22.11 所示。

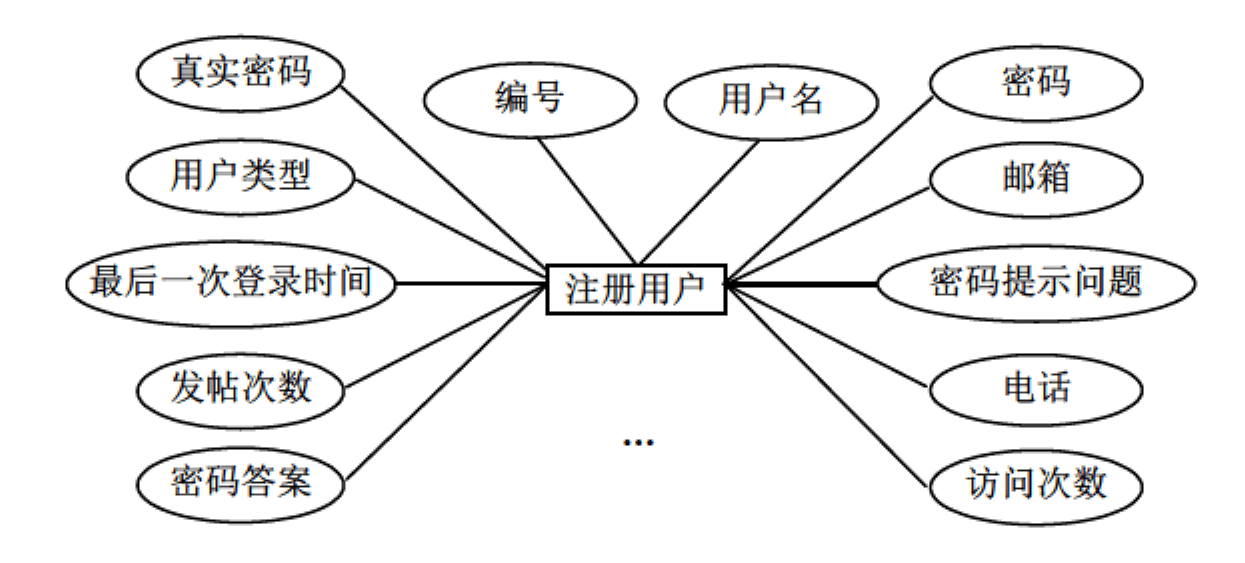


图 22.10　注册用户实体 E-R 图

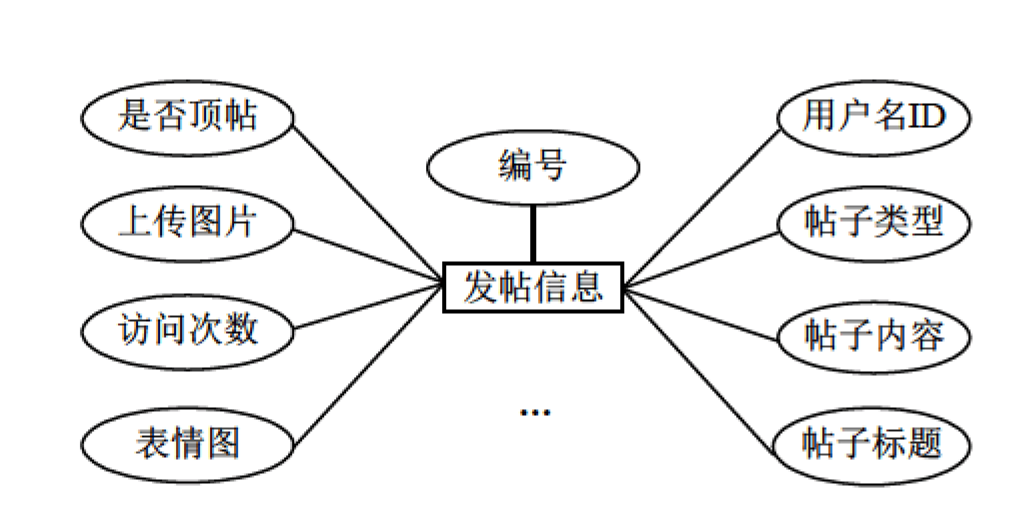
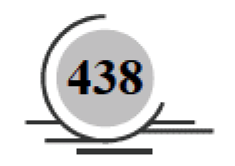


图 22.11　发帖信息实体的 E-R 图



3．订单信息实体

订单信息实体存储用户在线购买时填写的订单信息，包括编号、用户名、性别、家庭地址、邮政编码、QQ 号码、邮箱、手机号码、电话号码、收货方式、邮资、产品金额、订单时间、订单号和选择城市等属性。订单信息实体的 E-R 图如图 22.12 所示。

22.2.3　数据库逻辑设计

本项目中创建数据库和数据表使用的是 phpMyAdmin 图形化管理工具。下面将介绍数据库和数据表的创建方法，以及在创建过程中需要注意的一些问题。

1．数据库的创建

打开 phpMyAdmin 图形化管理工具的主页，首先在文本框中输入要创建的数据库的名称（如 db_bcty365），然后在下拉列表框中选择要使用的字符编码格式，这里使用的是 gb2312_chinese_ci，如图 22.13 所示。最后单击“创建”按钮，数据库创建成功。

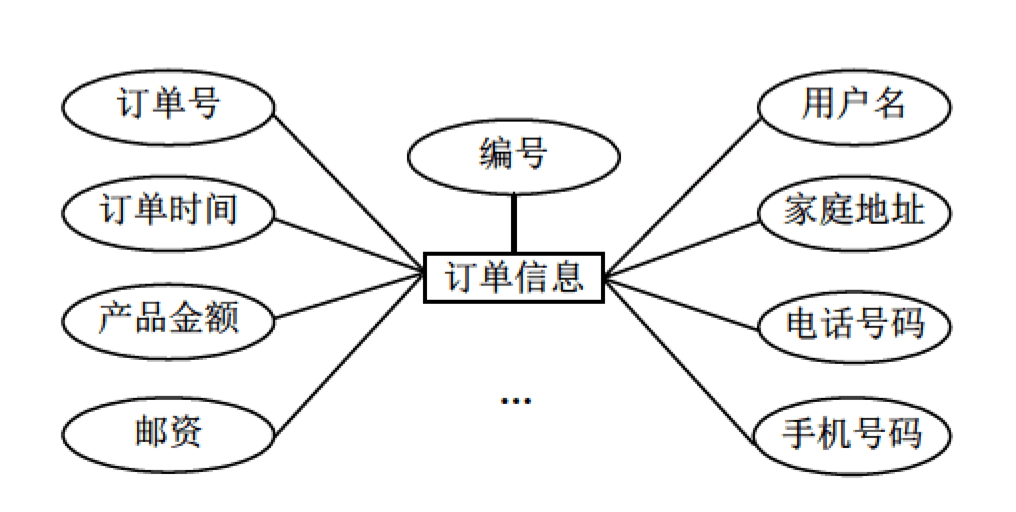


图 22.12　订单信息实体 E-R 图

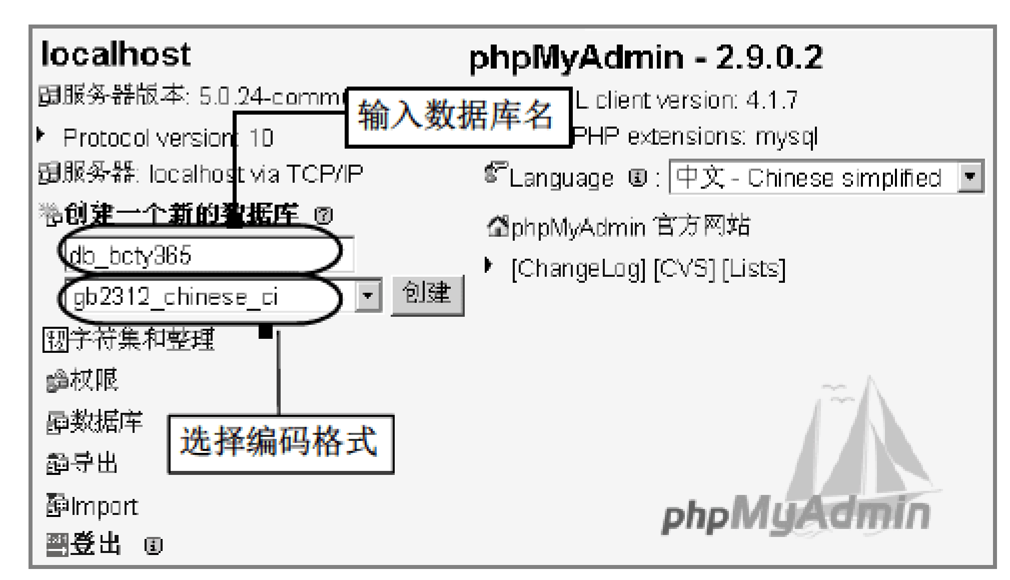


图 22.13　phpMyAdmin 管理界面

技巧 创建数据库的过程中，尽量使用与程序内容贴切的英文字符进行命名，有助于对数据库的理解。如果使用 AppServ 配置 PHP 开发环境，那么在创建数据库时不需要指定编码的格式，默认值为 gb2312_chinese_ci；如果自行配置开发环境，那么就要指定编码格式为 gb2312_chinese_ci，否则创建数据库的编码格式为 latin1_swedish_ci，将导致数据库中数据出现乱码。

2．创建数据表

在成功创建数据库后，接下来就是创建数据表，这里以 tb_bb 编程词典版本信息表为例，讲解如何创建数据表，以及在创建数据表的过程中需要注意哪些问题。这里创建一个名字为 tb_bb 的数据表，包括 3 个字段，如图 22.14 所示。

单击“执行”按钮后，进入到如图 22.15 所示的添加字段信息的页面中，在此处对字段进行详细设置，包括字段名、数据类型、长度/值、属性、默认值、额外、主键和索引等。

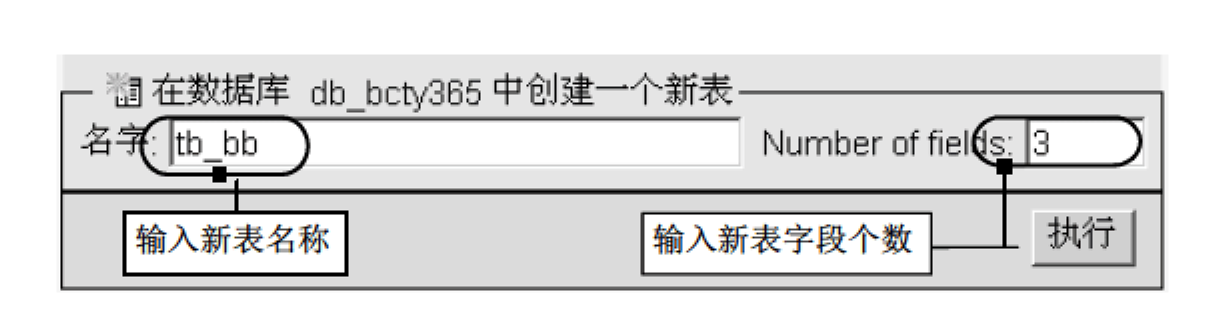


图 22.14　创建 tb_bb 数据表

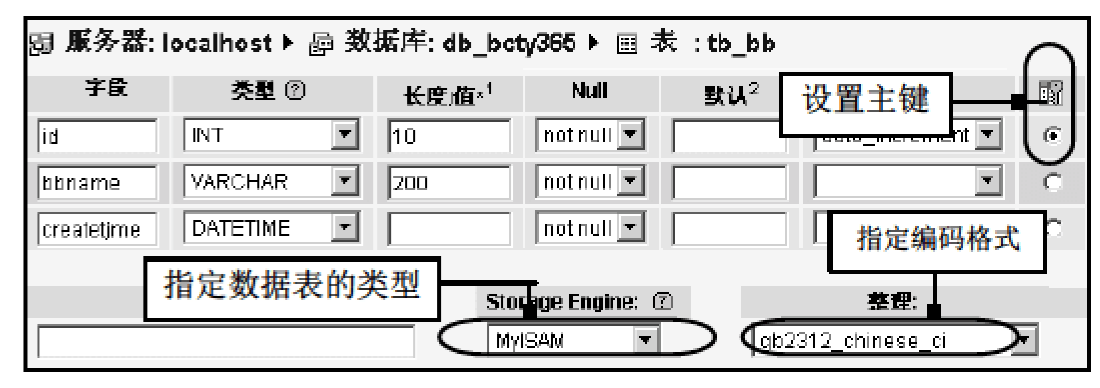
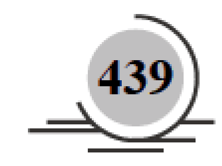


图 22.15　添加数据表中的字段信息



技巧 创建数据库中的数据表时，字段名的设计要尽量与数据表的内容相符合，这样有助于程序后期维护和修改工作的进行，能够直观地看出数据表的作用。

如果使用 AppServ 配置 PHP 开发环境，那么在创建数据表时不需要指定数据表类型和编码格式；如果使用自行配置的 PHP 开发环境，那么就要指定数据表的类型为 MyISAM，字符的编码格式为 gb2312_chinese_ci，否则创建的数据表类型为 InnoDB，而编码格式为 latin1_swedish_ci，这将导致该数据表中的数据复制到其他计算机上不可用，并且数据表中的数据出现乱码。

在创建数据表的过程中，一定要为数据表指定一个主键，它是数据表唯一的标识。

掌握数据表的创建方法后，就可以自行创建数据表。由于本项目中涉及的数据表有 17 个之多，这里不能对每个数据表的功能设计进行一一介绍，所以只给出几个重要的数据表的设计效果图供广大读者参考，其他数据表请参见本书附带的光盘。

（1）tb_user（注册用户信息表）

注册用户信息表主要用于存储本社区中会员的个人信息。该数据表的结构如图 22.16 所示。

服务器: localhost ▸ 数据库: db_bcty365 ▸ 表 : tb_user

字段	类型	整理	属性	Null	默认	额外	说明
id	int(8)			否		auto_increment	自动编号ID
usernc	varchar(50)	gb2312_chinese_ci		是	NULL		注册用户名
truename	varchar(50)	gb2312_chinese_ci		是	NULL		真实姓名
pwd	varchar(50)	gb2312_chinese_ci		是	NULL		注册密码
email	varchar(50)	gb2312_chinese_ci		是	NULL		有效邮箱地址
sex	varchar(2)	gb2312_chinese_ci		是	NULL		性别
tel	varchar(20)	gb2312_chinese_ci		是	NULL		联系电话
qq	varchar(20)	gb2312_chinese_ci		是	NULL		qq号码
address	varchar(100)	gb2312_chinese_ci		是	NULL		联系地址
logintimes	int(8)			否			访问次数
regtime	datetime			否			注册时间
lastlogintime	datetime			否			最后一次登录时间
ip	varchar(20)	gb2312_chinese_ci		否			IP地址
yb	varchar(20)	gb2312_chinese_ci		是	NULL		邮政编码
usertype	int(2)			否			用户类型
question	varchar(200)	gb2312_chinese_ci		否			密码提示问题
answer	varchar(200)	gb2312_chinese_ci		否			密码提示答案
truepwd	varchar(200)	gb2312_chinese_ci		否			真实密码
photo	varchar(50)	gb2312_chinese_ci		否			表情图
pubtimes	int(4)			是	0		发帖次数

图 22.16　注册用户信息表结构

（2）tb_reply（论坛回帖信息表）

论坛回帖信息表主要用于存储登录会员在本社区中回复帖子的信息。该数据表的结构如图 22.17 所示。

（3）tb_bccd（编程词典信息表）

编程词典信息表主要用于存储本社区在线订购模块中出售的编程词典的基本信息。该数据表的结构如图 22.18 所示。

服务器: localhost ▸ 数据库: db_bcty365 ▸ 表 : tb_reply

字段	类型	整理	属性	Null	默认	额外	说明
id	int(8)			否		auto_increment	自动编号ID
userid	int(8)			否	0		注册用户ID
bbsid	int(8)			否	0		发布帖子表ID
title	varchar(200)	gb2312_chinese_ci		是	NULL		回复主题
content	mediumtext	gb2312_chinese_ci		是	NULL		回复内容
createtime	datetime			是	NULL		回复时间
mark	int(2)			是	NULL		回复记录
photo	varchar(80)	gb2312_chinese_ci		是	NULL		回复图片

图 22.17　论坛回帖信息表结构



服务器: localhost ▸ 数据库: db_bcty365 ▸ 表 : tb_bccd

字段	类型	整理	属性	Null	默认	额外	说明
id	int(8)			否		auto_increment	自动编号ID
bccdname	varchar(200)	gb2312_chinese_ci		否			编程词典名称
owner	varchar(100)	gb2312_chinese_ci		否			开发者
typeid	varchar(50)	gb2312_chinese_ci		否			版本类型
content	mediumtext	gb2312_chinese_ci		否			编程词典简介
samepart	mediumtext	gb2312_chinese_ci		否			软件共同点
addtime	datetime			否			开发时间
imageaddress	varchar(100)	gb2312_chinese_ci		否			界面存储地址
bbid	int(8)			是	NULL		所属版本的ID
price	float			是	NULL		价格

图 22.18　编程词典信息表结构



22.3　前台首页设计

视频讲解：光盘\TM\Video\第 22 章\前台首页设计.exe

当今时代，很多人都十分重视事物的第一印象，在网络中更是如此，网站给人的第一印象如果不好，那么就会有很多人不去浏览该网站，无论网站的内容是否丰富。可以说网站首页设计的成功与否直接影响着整个网站的发展。

22.3.1　前台首页概述

网站首页是整个网站的脸面，既要突出企业的形象，又要展示出网站强大的功能。如果网站首页设计得非常成功，那无疑为整个网站的成功增添了一个砝码。BCTY365 网上社区首页的设计以企业的品牌形象为基础，全力打造网站的整体功能，重点推出企业的软件产品。具体内容如下所示。

- ☑ 网站菜单导航：包括首页、技术支持、在线订购、社区论坛、软件下载、升级下载、购买须知和联系我们。
- ☑ 用户注册和登录模块：实现用户注册、会员登录、找回密码和修改密码的功能。
- ☑ 网站公告：主要用于发布社区中的一些新消息和重大事件。
- ☑ 编程词典模块：推广企业的软件产品。
- ☑ 软件下载模块：展示企业提供的试用版和免费的软件产品。
- ☑ 常见问题模块：列举出编程中常见问题及其解决方案。
- ☑ 社区论坛模块：浏览社区论坛中的部分帖子。
- ☑ 升级下载模块：提供一些软件的升级版本下载。

上述就是 BCTY365 网上社区首页体现的内容，为了更加直观地了解网上社区首页的设计，这里先预览一下社区首页，该首页在本书光盘中的路径为光盘\TM\22\index.php，如图 22.19 所示。



图 22.19　BCTY365 网上社区首页

BCTY365 网上社区首页的设计看上去很复杂，由很多版块组成，但实现的过程非常简单。总体架构使用一个 2 行 3 列表格和一个 3 行 3 列表格，将其分割成不同的版块，然后使用脚本语句从数据库中读取数据，最后将数据循环输出到页面中，其中网站的头尾文件使用 include 包含语句调用。首页的框架结构如图 22.20 所示。



图 22.20　网站首页的框架结构

22.3.2　公告信息的滚动输出技术

作为网站首页，不一定要具有特殊的功能，而应该简洁、鲜明，以突出企业形象、展示网站的功能为主。即使使用的是一个静态页面，只要能够将内容表达全面、完整，那么这个首页设计也是非常成功的。

在本项目首页的设计中，应用到一个文字循环滚动的技术，通过该技术来输出社区中发布的公告信息。该技术是通过 JavaScript 脚本和<div>标签来共同实现的，其实现的原理是：首先创建一个<div>标签，然后在<div>标签中输出公告信息，最后通过 JavaScript 来对<div>标签进行操作，实现<div>标签中内容的滚动输出。该技术的实现在 index.php 页中完成。其中使用的 JavaScript 脚本的代码如下：

（代码位置：光盘\TM\Instance\22\index.php）

```
<script language="JavaScript">
    marqueesHeight=222;                                          //定义输出标签的高度
    stopscroll=false;                                            //定义 stopscroll 的默认值为 false
    with(marquees){                                              //编辑 marquees 标签的属性
        style.width=0;                                           //定义初始宽为 0
        style.height=marqueesHeight;                             //定义 marquees 标签的高为 222
        style.overflowX="visible";                               //定义值为显示
        style.overflowY="hidden";                                //定义值为隐藏
        noWrap=true;
        onmouseover=new Function("stopscroll=true");             //当光标经过时执行 stopscroll=true
        onmouseout=new Function("stopscroll=false");             //当光标离开时执行 stopscroll=false
    }
    //创建一个新的 div"templayer"与 div"marquees"进行连接，实现不间断地循环输出内容
    document.write('<div id="templayer" style="position:absolute;z-index:1;visibility:hidden"></div>');
```

```
	preTop=0; currentTop=0;
	function init(){
		templayer.innerHTML="";					//设置 templayer 的初始值为空
		while(templayer.offsetHeight<marqueesHeight){		//判断当 templayer 的高度小于 marquees 的高度时
			templayer.innerHTML+=marquees.innerHTML;	//将 templayer 的值赋给 marquees
		}
		marquees.innerHTML=templayer.innerHTML+templayer.innerHTML;	//将 templayer 的值累加
		setInterval("scrollup()",50);				//间隔 50 毫秒执行一次 scrollup()函数
	}
	function scrollup(){						//实现滚动输出
		if(stopscroll==true) return;				//判断如果 stopscroll==true，不执行循环
		preTop=marquees.scrollTop;
		marquees.scrollTop+=1;
		if(preTop==marquees.scrollTop){
			marquees.scrollTop=templayer.offsetHeight-marqueesHeight;
			marquees.scrollTop+=1;
		}
	}
</script>
```

在<div>标签中，主要是输出数据库中存储的公告信息，并且对输出的信息进行截取和替换，规范输出的内容。<div>标签中的程序代码如下：

```
<div id=marquees >  <!-->创建一个 div 标签<!-->
<table width="200" height="25" border="0" align="center"    cellpadding="0" cellspacing="0">
	<?php								//从数据库中读取公告数据
		$sql=mysql_query("select id,title,createtime from tb_tell order by createtime desc limit 0,10",$conn);
		$info=mysql_fetch_array($sql);
		if($info==false){					//判断当$info==false 时执行下面的内容
	?>
	<tr>
	<td height="25"><div align="center"><a href="#" class="a4">本站暂无公告发布! </a></div></td>
	</tr>
	<?php	}else{
		$i=1;							//定义变量$i=1
		do{							//执行 do…while 循环语句
	?>
	<tr>
	<td height="25" style="padding:4">
			<a href="tellinfo.php?id=<?php echo $info['id'];?>" class="a1">
	<?php
		if($i==1){						//当$i==1 时，将输出的内容设置为红色
			echo "<font color=red>";
		}
		echo $i.". ";
		echo unhtml(msubstr($info['title'],0,50));		//应用自定义函数对输出的内容进行控制
		if(strlen($info['title'])>50){			//当输出内容的长度超过 50 个字符时用“…”代替
			echo " ...";
		}
		echo "(".str_replace("-","/",$info['createtime']).")";	//将输出的公告时间中的“-”用“/”替代
		if($i==1){  echo "</font>"; }
	?>		</a>
		</td>
	</tr>
	<?php
		$i++;
```

```
            }while($info=mysql_fetch_array($sql));              //do…while 循环语句结束
            }
        ?>
</table>
    </div>
```

在首页中使用滚动条是一个不错的方法，添加滚动条可以增加网页的动态效果，提高网页的观赏性，而且不会影响网页的浏览速度。

22.3.3 前台首页的实现过程

开发网站首页主要就是连接数据库，然后从数据库中读取数据，最后应用循环语句将数据库中数据输出到前台页面。由于使用的代码较多，而且多数都是重复使用，所以这里只给出首页中公告发布模块的代码。

公告发布模块主要实现从数据库中读取公告数据，将数据在首页中滚动输出，并且对公告信息的长度进行控制，保证内容的整齐、规范。详细代码可以参考本书光盘中 TM\22\bcty365\index.php 文件。index.php 文件的部分代码如下：

（代码位置：光盘\TM\Instance\22\index.php）

```
<?php include_once("top.php"); ?>                                //获取头部文件
…//省略部分代码
<?php
      //读取数据库中公告表中的数据
      $sql=mysql_query("select id,title,createtime from tb_tell order by createtime desc limit 0,10",$conn);
      $info=mysql_fetch_array($sql);                             //执行读取数据表中数据的语句
      if($info==false){                                          //如果返回值为空则执行下面的语句
?>
<tr>
      <td height="25"><div align="center"><a href="#" class="a4">本站暂无公告发布! </a></div></td>
</tr>
<?php
      }else{                                                     //如果返回值不为空则执行下面的 do…while 循环语句
            $i=1;
            do{
?>
<tr>
      <td height="25" style="padding:4"><a href="tellinfo.php?id=<?php echo $info['id'];?>" class="a1">
<?php
      if($i==1){                                                 //当变量$i=1 时，输出的内容以红色字体显示
            echo "<font color=red>";
      }
      echo $i.". ";
❶     echo unhtml(msubstr($info['title'],0,50));
❷     if(strlen($info['title'])>50){                             //如果标题长度大于 50 个字符，则以省略号代替
            echo " ...";
      }
❸     echo "(".str_replace("-","/",$info['createtime']).")";     //输出公告发布的时间，并且将其中的"/"使用"-"替换
      if($i==1){
            echo "</font>";
      }
?>
            </a>
      </td>
</tr>
```



```
<?php
$i++;                                          //变量自加 1
}while($info=mysql_fetch_array($sql));         //do…while 循环语句结束
}
?>
…                                              //省略了部分代码
<?php
include_once("bottom.php");                    //调用网站的尾文件
?>
```

说明

❶ 应用 unhtml()和 msubstr()自定义函数去除输出字符串中的空格和控制输出字符串的长度。
❷ strlen()函数用于获取指定字符串的长度。
❸ str_replace()函数用于实现字符串的替换。

22.4 注册模块设计

视频讲解：光盘\TM\Video\第 22 章\注册模块设计.exe

22.4.1 注册模块概述

为了更好地与广大网民朋友进行交流和沟通，BCTY365 网上社区系统中创建了一个会员注册模块。通过会员注册模块，可以有效地对用户信息进行采集，并将合法的用户信息保存到指定的数据表中，实现与用户的长期沟通和交流。既然设置了会员注册模块，那么在系统中就要为会员提供一些特殊的权限。在本系统中注册为会员可以拥有如下权限：在本社区的论坛中发布和回复帖子、在技术支持模块中发表留言、在升级下载模块中下载软件升级包等，而且可以修改和找回密码。用户注册模块的运行结果如图 22.21 所示。



图 22.21　用户注册模块的运行结果

22.4.2 通过 JavaScript 脚本验证表单元素

在会员注册模块中，必不可少的功能就是要对用户输入的信息进行判断，首先判断用户填写的注册信

息中哪些是必须填写的、哪些可以不填写，然后进一步判断输入的信息是否合理、合法，如判断输入的邮编格式是否正确、邮箱格式是否正确等。对表单中提交数据进行判断最常用的办法就是使用 JavaScript 脚本，也可以使用正则表达式。下面讲解在本模块中是如何通过 JavaScript 脚本实现表单提交数据验证。

操作原理是：在 form 表单中调用 onsubmit 事件，通过该事件调用指定的 JavaScript 脚本，执行 chkinput() 自定义函数，实现对表单中提交数据的验证。在 JavaScript 脚本中，对表单中提交数据进行判断，判断输入的内容是否为空、内容的格式是否正确，如果正确则继续执行，否则将弹出提示对话框，并将鼠标的焦点指定到出错的位置。具体的 JavaScript 脚本代码如下：

```
<script language="JavaScript" type="text/javascript">
    function chkinput(form){                                    //定义一个函数
        if(form.tel.value==""){                                 //判断 tel 文本框中的值是否为空
            alert("请填写联系电话!");                            //如果为空则输出“请填写联系电话!”
            form.tel.select();                                  //返回到 tel 文本框
            return(false);
        }
        if(form.email.value==""){                               //判断 email 文本框的值是否为空
        alert("请输入 E-mail 地址!");                            //如果为空则输出“请输入 E-mail 地址!”
            form.email.select();                                //返回到 email 文本框
            return(false);
        }
        var i=form.email.value.indexOf("@");
        var j=form.email.value.indexOf(".");
        //进一步判断邮箱的格式是否正确，是否包含“@”和“.”
        if((i<0)||(i-j>0)||(j<0)){
            alert("请输入正确的 E-mail 地址!");
            form.email.select();                                //返回到 email 文本框
            return(false);
        }
     return(true);                                              //提交表单
    }
</script>
```

上述代码中，只是列举了 JavaScript 中的部分内容，并且在对电话号码进行判断时，只是判断其是否为空，没有进一步判断电话号码的格式是否正确。如果想要更加准确地判断电话号码的格式是否正确，可以采用下面的方法：通过正则表达式的 preg_match()函数，在表单提交处理页中对电话号码进行判断。

preg_match()函数的语法格式如下：

```
int preg_match ( string pattern, string subject [, array matches [, int flags]] )
```

preg_match()函数的参数说明如表 22.1 所示。

表 22.1　preg_match()函数的参数说明

参　　数	说　　明
pattern	必选参数。需要匹配的正则表达式
subject	必选参数。输入的字符串
matches	可选参数。输出的搜索结果的数组，如$out[0]将包含与整个模式匹配的结果，$out[1]将包含与第一个捕获的括号中的子模式所匹配的结果，依次类推
flags	可选参数。若设为 PREG_OFFSET_CAPTURE，对每个出现的匹配结果也同时返回其附属的字符串偏移量，本标记自 PHP 4.3.0 起可用

通过 preg_match()函数判断电话号码的格式是否正确的方法如下。

首先定义一个用于判断电话号码格式的正则表达式。代码如下：

```
/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$/
```



正则表达式的功能分析如下：使用“^”和“$”对字符串进行边界的限制，对区号从字符串的开始进行匹配，对其他号码从字符串的末尾开始进行匹配；将“()”中的内容作为一个原子使用；使用“\d”来匹配一个数字，区号为 3 或 4 个数字，其他数字为 7 或 8 个；使用“{}”来对前字符进行重复匹配；使用“|”对匹配的模式进行选择，分成 3 个模式。

然后将该正则表达式应用到 preg_match()函数中，对表单提交的电话号码进行判断，如果正确则继续执行，否则弹出提示信息，并返回到表单提交页。代码如下：

```
<?php
    $tel="0431-84978981";                                        //定义一个电话号码的变量
    if(preg_match("/^(\d{3}-)(\d{8})$|^(\d{4}-)(\d{7})$|^(\d{4}-)(\d{8})$|^(\d{11})$/",$tel,$counts)){
           echo "您输入的电话号码格式正确!";                     //输出字符串
    }else{
          echo "<script>alert('您输入的电话号码的格式不正确!!');history.back()</script>";
    }
    ?>
```

22.4.3　注册模块的实现过程

注册模块的实现过程非常简单，首先阅读注册服务条款，然后填写注册的用户名和密码，提交后由系统判断输入的用户名是否被占用，如果未被占用则可以继续注册，填写详细的注册信息，将数据提交到表单处理页进行处理，最后将用户注册的信息保存到指定的数据表中。用户注册模块的实现过程主要由 3 个文件完成：register.php 文件用于输出注册服务条款，以及填写注册的用户名和密码，并且判断注册的用户名是否被占用；getuserinfo.php 文件用于填写详细的注册信息，并且在表单中应用数字验证码技术；savereginfo.php 文件用于对表单中提交的数据进行处理，将数据保存到指定的数据表中。

在 savereginfo.php 文件中，首先连接数据库，然后获取表单中提交的数据，并且判断提交的用户名是否被占用，最后将提交的数据进行处理，并将数据保存到指定的数据表中。程序代码如下：

（代码位置：光盘\TM\Instance\22\saverreginfo.php）

```
<?php session_start();                                              //初始化 session 变量
include_once("conn/conn.php");                                      //连接数据库
$usernc=trim($_POST['usernc']);                                     //获取注册的用户名
//判断指定的用户名是否存在
$sql=mysql_query("select usernc from tb_user where usernc='".$usernc."'",$conn);
$info=mysql_fetch_array($sql);                                      //按指定条件检索数据信息
if($info!=false){                                                   //如果查询结果不为空，则执行以下操作
      echo "<script language='javascript'>alert('对不起，该昵称已被其他用户使用!');history.back();</script>";
      exit;
}
$xym=trim($_POST['xym']);                                           //去除变量两边的空格
$num=$_POST['num'];                                                 //接收变量值
if(strval($xym)!=strval($num)){
      echo "<script>alert('验证码输入错误!');window.location.href='register.php';</script>";
      exit;}
//对表单提交的数据进行处理
$truepwd=trim($_POST['pwd1']);                                      //获取真实密码
$pwd=md5($truepwd);                                                 //获取加密密码
$truename=trim($_POST['truename']);                                 //获取真实姓名
$email=trim($_POST['email']);                                       //获取邮箱地址
$sex=$_POST['sex'];                                                 //获取性别
$tel=trim($_POST['tel']);                                           //获取电话
```

```
$yb=trim($_POST['yb']);                                     //获取邮政编码
$qq=trim($_POST['qq']);                                     //获取 QQ
$address=trim($_POST['address']);                           //获取地址
$question=trim($_POST['question']);                         //获取提示问题
$answer=trim($_POST['answer']);                             //获取问题答案
$ip=getenv("REMOTE_ADDR");                                  //获取客户端的 IP
$logintimes=1;                                              //指定访问次数
$regtime=date("Y-m-j H:i:s");                               //获取时间
$lastlogintime=$regtime;
$usertype=0;                                                //指定用户类型，默认为 0
$photo=$_POST["photo"];                                     //获取头像
//将表单中提交的数据存储到数据表中
if(mysql_query("insert into
tb_user(usernc,truename,pwd,email,sex,tel,qq,address,logintimes,regtime,lastlogintime,ip,usertype,yb,question
,answer,truepwd,photo)
values('$usernc','$truename','$pwd','$email','$sex','$tel','$qq','$address','$logintimes','$regtime','$lastlogintime','
$ip','$usertype','$yb','$question','$answer','$truepwd','$photo')",$conn)){
    session_register("unc");
    $_SESSION["unc"]=$usernc;
    echo "<script>alert('注册成功!');window.location.href='index.php';</script>";
}else{                                                      //如果添加操作失败，则给出提示
    echo "<script language='javascript'>alert('对不起,注册失败!');history.back();</script>";
    exit;                                                   //退出
}
?>
```

22.5 技术支持模块设计

视频讲解：光盘\TM\Video\第 22 章\技术支持模块设计.exe

技术支持模块主要是从浏览者的角度进行设计，存储大量技术问题的解决方案数据，为浏览者查阅提供方便，而且设计一个企业与客户沟通的平台，能够随时了解客户或者会员的意见和需求。

22.5.1 技术支持模块概述

技术支持模块主要由 3 个子模块组成，包括常见问题模块、客户反馈模块和联系方式模块。常见问题模块主要用于展示编程中一些常见问题的解决方案或者方法，为浏览者解决编程中的疑难问题提供方便；客户反馈模块主要用于收集和获取来自客户的需求和意见；联系方式模块主要用于展示企业的形象和具体的联系方式。

22.5.2 分页技术

技术支持模块中，在对常见问题的解决方案数据进行输出时，使用了分页处理技术，该技术的设计思路是：从数据库中读取数据，获取数据总量，在每页中显示 20 条数据，根据数据总量和每页显示的条数对数据进行分页处理，计算出有多少页和当前显示的页码，实现首页、上一页、下一页和尾页之间的页面跳转。具体的设计思路可以参考 cjwt.php 文件中的代码注释和代码贴士。cjwt.php 文件的程序代码如下：

（代码位置：光盘\TM\Instance\22\cjwt.php）

```
<?php
    $sql=mysql_query("select count(*) as total from tb_cjwt",$conn);    //读取数据库中数据
    $info=mysql_fetch_array($sql);                                       //返回数据
❶   $total=$info['total'];
    //判断字段 total 是否为空，为空则执行下面的内容
    if($total==0){
?>
    <tr>
    <td height="22" colspan="2"><div align="center">对不起，暂无常见问题!</div></td>
   </tr>
<?php
    }else{                                                      //如果不为空，则执行下面的内容
❷       if(!isset($_GET["page"]) || !is_numeric($_GET["page"])){ //判断$_GET 获取的 page 的值是否存在
❸           $page=1;                                            //如果不存在，则设置变量的值为 1
        }else{
❹           $page=intval($_GET["page"]);                        //如果存在，则获取变量$_GET 的值
        }
        //设置变量$pagesize，每页显示的数据量为 20 条
❺       $pagesize=20;
        if($total%$pagesize==0){                                //如果变量的值为 0
❻           $pagecount=intval($total/$pagesize);                //获取变量的整数值
        }else{
❼           $pagecount=ceil($total/$pagesize);                  //如果不为 0，则获取实际的整数值
        }
        //读取数据库中数据，按照时间进行降幂排列
        $sql=mysql_query("select * from tb_cjwt order by createtime desc limit ".($page-1)*$pagesize.",$pagesize
",$conn);
        while($info=mysql_fetch_array($sql)){
?>
… //省略部分代码
<table width="600" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
    <tr>
    <td width="479"><div align="left">  共有常见问题 <?php echo $total;?> 条 
每 页 显 示  <?php  echo  $pagesize;?>  条   第  <?php  echo  $page;?>  页 / 共
 <?php echo $pagecount;?> 页</div></td>
      <td width="289"><div align="right">
❽           <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=1" class="a1">首页</a> 
            <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=<?php
                if($page>1)                                     //判断如果页码大于 1
                    echo $page-1;                               //输出前一页
                else
                    echo 1;
                    ?>" class="a1">上一页</a> 
            <a href="<?php echo $_SERVER["PHP_SELF"]?>?page=<?php
            if($page<$pagecount)                                //如果页码小于总页数
                echo $page+1;
            else
                echo $pagecount;                                //输出下一页
                    ?>" class="a1">下一页</a> 
            <a   href="<?php   echo   $_SERVER["PHP_SELF"]?>?page=<?php   echo   $pagecount;?>"
class="a1">尾页</a>  </div>
        </td>
      </tr>
    </table>
```

说明

❶ $total：为数据库中数据总的记录数。
❷ isset()：检测变量是否已经设置；is_numeric()：检测变量是否为数字或者数字字符串。
❸ $page：变量为页码中的第几页。
❹ intval()：获取变量的整数值。
❺ $pagesize：表示在每页中显示多少条数据。
❻ $pagecount：表示所有的数据可以分成多少页。
❼ ceil()：获取变量中的整数值，这里用于获取页码的整数值。
❽ $_SERVER["PHP_SELF"]：服务器变量，这里用于获取网页的链接地址。

22.5.3　常见问题模块的实现过程

常见问题子模块实现的主要功能是，展示出数据库中存储的有关编程中遇到的常见问题及解决方案。其运行结果如图 22.22 所示。

该模块由两个文件组成，一个是 cjwt.php 文件，用于存储创建的问题数据；另一个是 lookcjwt.php 文件，用于输出 cjwt.php 文件中对应问题的详细介绍和解决方案。其代码如下：

（代码位置：光盘\TM\Instance\22\lookcjwt.php）

```
<?php
include_once("conn/conn.php");                                    //与数据库建立连接
include_once("function.php");                                     //调用自定义函数
//读取 tb_cjwt 表中的数据，条件为 id="$_GET['id']"
$sql=mysql_query("select * from tb_cjwt where id='".$_GET["id"]."'",$conn);
$info=mysql_fetch_array($sql);
?>
<table width="635" height="100" border="0" align="center" cellpadding="0" cellspacing="0">
    <tr>
    <td width="94" height="30"><div align="center"><strong>问  题：</strong></div></td>
        <td width="541"><?php echo unhtml($info["question"]); ?></td>          //输出问题的详细内容
    </tr>
    <tr>
        <td height="70"><div align="center"><strong>解  答：</strong></div></td>
        <td height="70"> <?php echo unhtml($info["answer"]);?></td>  //输出问题的解决方案
    </tr>
</table>
```

22.5.4　客户反馈模块的实现过程

客户反馈子模块为客户提供一个反馈意见和提出要求的平台，并且将提交的信息存储到数据库中。其运行结果如图 22.23 所示。

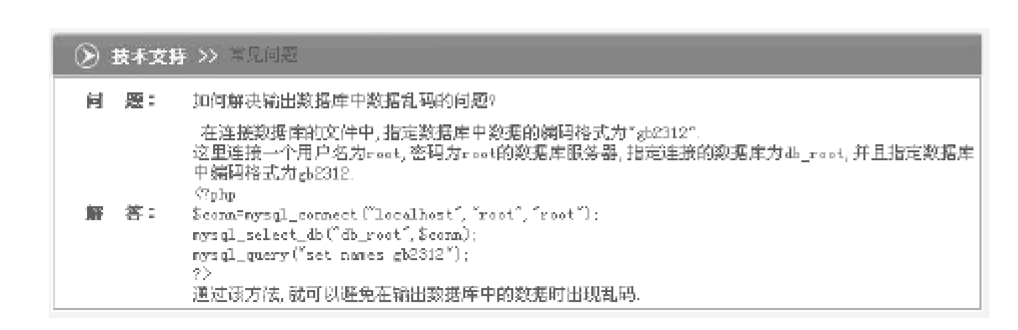


图 22.22　常见问题模块的运行结果

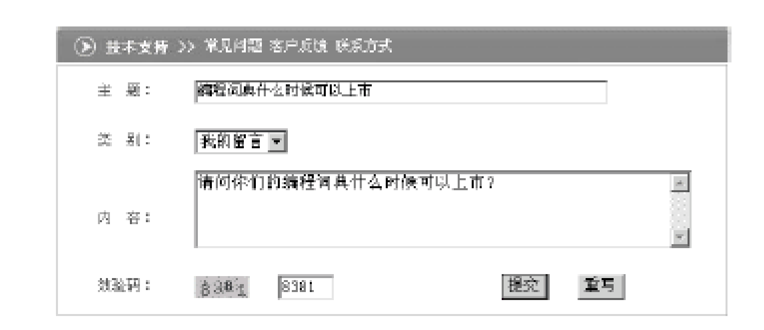


图 22.23　客户反馈模块的运行结果



该功能只对本网站中的会员开通，即只有以会员身份登录的用户才具有反馈信息的权限，其中在对提交表单的细节处理上使用 JavaScript 脚本来验证表单中的值是否为空，而且还使用数字验证码技术。saveleaveword.php 文件用于对表单中提交的数据进行处理，将数据存储到数据库中。其代码如下：

（代码位置：光盘\TM\Instance\22\saveleaveword.php）

```
<?php session_start();                                    //初始化一个 session 变量
$xym=$_POST['xym'];                                       //获取$_POST 提交的值
if($xym!=$_SESSION["autonum"]){                           //判断验证码是否正确
    echo "<script>alert('效验码输入错误!');history.back();</script>";
    exit;
}
$title=$_POST["title"];                                   //获取反馈信息的标题
$content=$_POST["content"];                               //获取反馈信息的内容
$type=$_POST["type"];                                     //获取反馈信息的类型
include_once("conn/conn.php");                            //与数据库建立连接
$sql=mysql_query("select id from tb_user where usernc='".$_SESSION["unc"]."'",$conn); //读取数据库中数据
$info=mysql_fetch_array($sql);                            //获取结果集中的数组
$userid=$info["id"];
//向数据库中添加数据
if(mysql_query("insert into tb_leaveword(userid,type,title,content,createtime)
values('$userid','$type','$title','$content','".date("Y-m-j H:i:s")."')",$conn)){
    echo "<script>alert('留言发表成功!');history.back();</script>";      //添加操作成功，给出提示信息
}else{
    echo "<script>alert('留言发表失败!');history.back();</script>";      //添加操作失败，给出提示信息
}
?>
```

22.6　在线订购模块设计

视频讲解：光盘\TM\Video\第 22 章\在线订购模块设计.exe

在线订购模块的功能是实现在线购买企业推出的软件产品，其操作主要通过购物车和订单管理来实现。

22.6.1　在线订购模块概述

在线订购模块的功能对所有访问网站的人开放，没有任何权限限制。其操作流程如图 22.24 所示。

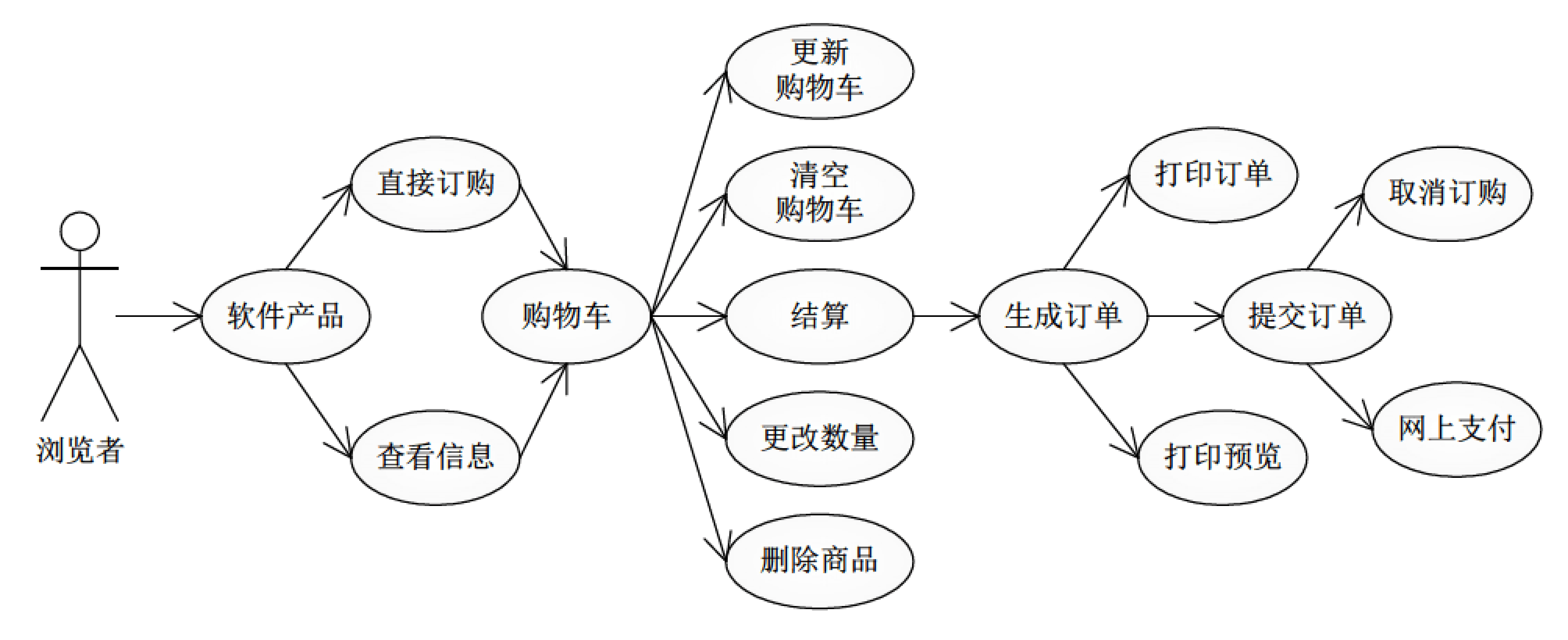


图 22.24　在线订购模块的操作流程

22.6.2 订单的预览及打印技术

在线订购管理模块中，不可缺少的一项内容就是对订单进行打印。下面就来讲解一下订单打印功能的实现方法。在线订购管理模块中运用的是 WebBrowser 打印方法。WebBrowser 是 IE 内置的浏览器控件，无须用户下载。其优点是客户端独立完成打印目标文档的生成，减轻服务器负荷；缺点是源文档的分析操作复杂，并且要对源文档中要打印的内容进行约束。

下面介绍 WebBrowser 控件的具体参数，如表 22.2 所示。

表 22.2 WebBrowser 控件的具体参数及说明

参 数 名 称	说 明
document.all.WebBrowser.Execwb(7,1);	表示打印预览
document.all.WebBrowser.Execwb(6,1);	表示打印
document.all.WebBrowser.Execwb(6,6);	表示直接打印
document.all.WebBrowser.Execwb(8,1);	表示页面设置
document.all.WebBrowser.Execwb(1,1);	打开页面
document.all.WebBrowser.Execwb(2,1);	关闭所有打开的 IE 窗口
document.all.WebBrowser.Execwb(4,1);	保存网页
document.all.WebBrowser.Execwb(10,1);	查看页面属性
document.all.WebBrowser.Execwb(17,1);	全选
document.all.WebBrowser.Execwb(22,1);	刷新
document.all.WebBrowser.Execwb(45,1);	关闭窗体无提示

该技术的实现原理是：首先通过 onClick 事件调用一个 JavaScript 脚本，然后执行 openprintwindow()函数，将指定的变量值传递到订单打印页（printwindow.php）中，最后在订单打印页中实现打印及打印预览的功能。调用 JavaScript 脚本和执行 openprintwindow()函数在 shopping_dd.php 页中完成。其关键代码如下：

（代码位置：光盘\TM\Instance\22\shopping_dd.php）

```
<script language="javascript">
      function openprintwindow(x,y,z){                              //定义一个函数，获取传递的参数
      //通过 window 对象中的 open()方法打开一个新窗口，并设置其属性
❶     window.open("printwindow.php?ddno="+x+"&pv="+z,"newframe","top=200,left=200,width=635,
height="+(230+20*y)+",menubar=no,location=no,toolbar=no,scrollbars=no,status=no");
        }
      </script>
<!-- ---------通过 onclick 事件调用 JavaScript 脚本，传递参数值。当参数 z 的值为 p 时执行打印功能----------- -->
<td width="75"> <img src="images/bg_14(11).jpg" width="69" height="20"
❷      onclick="javascript:openprintwindow('<?php echo base64_decode($_GET["ddno"]);?>','<?php echo
$gnum;?>','p')" style="cursor:hand"/></td>
<!-- ------通过 onclick 事件调用 JavaScript 脚本，传递参数值。当参数 z 的值为 v 时执行打印预览功能--------- -->
<td width="90"><img src="images/bg_14(12).jpg" width="69" height="20"
onclick="javascript:openprintwindow('<?php echo base64_decode($_GET["ddno"]);?>','<?php echo
$gnum;?>','v')"
style="cursor:hand"/></td>
```



说明

❶ open()：打开一个窗口，可以设置其 URL、名称、大小、按钮以及其他属性。该方法的基本语法结构为：

window.open(url,name,features,replace)

参数 url 是要在新窗口中打开的文档的 URL 地址；参数 name 是要打开的窗口的名字，用 HTML 链接的 target 属性进行定位时会用到；参数 features 是一个用逗号分隔的字符串，列举窗口的特征；参数 replace 是一个可选的 Boolean 值，指出是否允许 URL 替换窗口的内容。

❷ onclick：表明某元素被鼠标单击触发的事件。

订单的打印和打印预览功能在 printwindow.php 页中完成，首先编写一个实现打印预览功能的 JavaScript 脚本，然后建立 HTML 的 object 标签，调用 WebBrowser 控件，最后获取变量传递的值，当变量的值为 p 时执行打印功能；当变量的值为 v 时执行打印预览的功能。printwindow.php 页的关键代码如下：

（代码位置：光盘\TM\Instance\22\printwindow.php）

```
<script>
     function printview(){                                          //定义一个函数
❶         document.all.WebBrowser1.ExecWB(7,1) ;                    //调用 WebBrowser 控件，实现打印预览
          window.close();                                           //关闭窗口
     }
</script>
<!-- ---------------------------------建立 HTML 的 object 标签，调用 WebBrowser 控件-------------------------- -->
<object ID='WebBrowser1' WIDTH=0 HEIGHT=0
CLASSID='CLSID:8856F961-340A-11D0-A96B-00C04FD705A2'></object>
<!--- -----------------------------------在 body 中调用 onload 事件，执行打印操作------------------------------------ -->
❷    <body topmargin="0" leftmargin="0" bottommargin="0" onLoad="
<?php
     if($_GET["pv"]=="p"){                                          //判断当变量值为 p 时，执行打印操作
?>
❸    window.print();
<?php
     }elseif($_GET["pv"]=="v"){                                     //判断当变量值为 v 时，执行打印预览操作
?>
     printview()
<?php }    ?>
">
```

说明

❶ document.all.WebBrowser1.ExecWB(7,1)：WebBrowser 控件中的参数，表示打印预览。

❷ onLoad：表明某对象已载入窗口。

❸ print()：window 对象中的一个方法，实现打印的功能。

订单打印操作的运行效果如图 22.25 所示。

22.6.3　购物车的实现过程

购物车的功能是临时存储用户选购的商品，用户可以对购物车中的商品进行添加、修改、删除和更新操作，也可以选择进行结算。其运行的效果如图 22.26 所示。

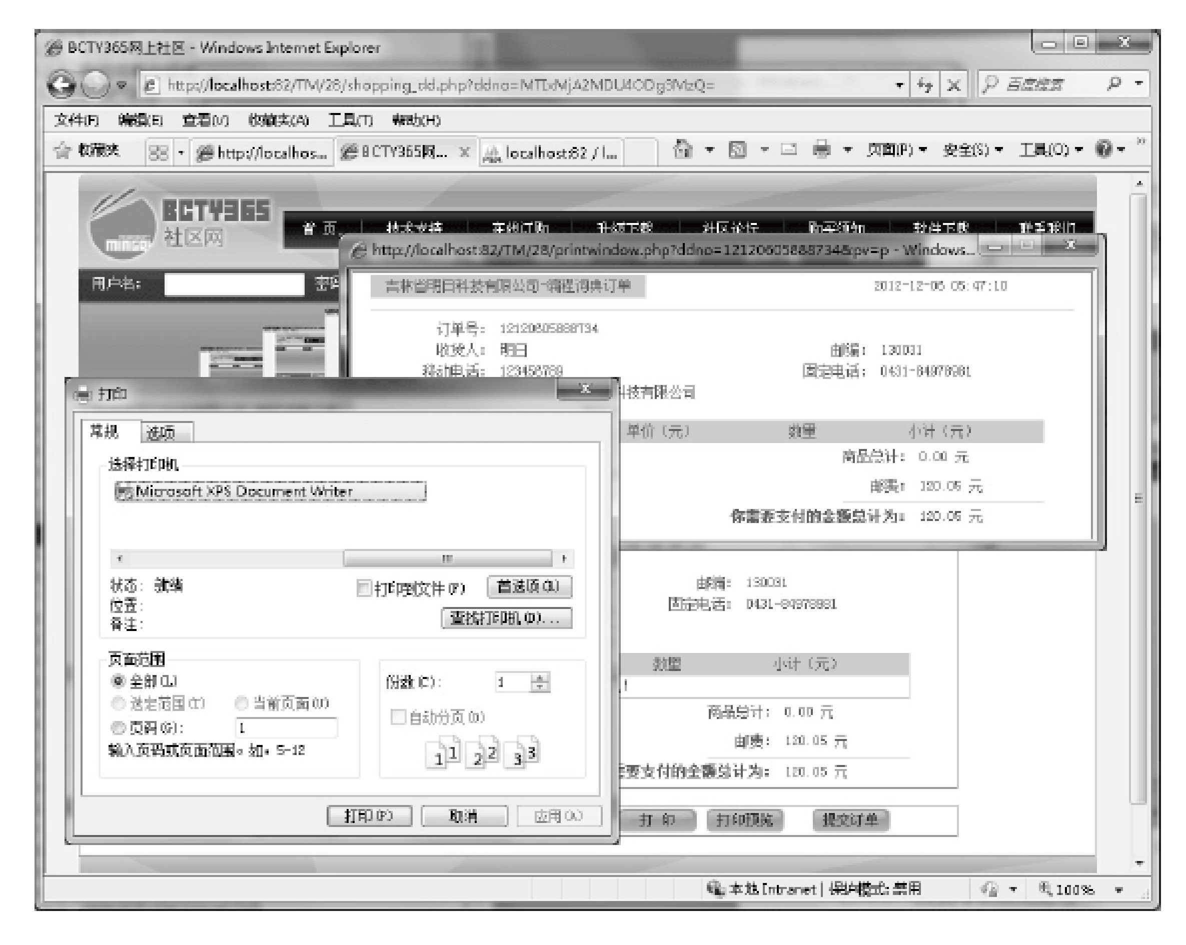


图 22.25　订单打印操作的运行效果



图 22.26　购物车的运行效果

购物车功能实现的第一步是为想要购买产品的用户分配一辆购物车，使其能够记录自己已经选购的产品。其工作的原理与超市中顾客使用购物车进行购物是相同的，只是这里使用的不是真正意义上的购物车，而是两个 session 变量，一个存储用户选购商品的 ID（$goodsid），另一个存储用户选购该商品的数量（$goodsnum）。如果用户在一次购物中选购多种不同类的商品，则使用“@”对不同类商品的不同 ID 和数量进行分隔。例如，用户选购的不同类商品 id 为 1、2、3，则 session 变量$goodsid 中存储的值为“1@2@3@”。其中同一种商品不能购买两次，如果想要购买多个同种产品，可以在购物车中更改购买商品的数量。在本项目中，购物车的分配功能通过 shopping_cart_first.php 文件来完成。shopping_cart_first.php 文件的代码如下：

（代码位置：光盘\TM\Instance\22\shopping_cart_first.php）

```
❶   <?php    session_start();                                    //初始化 session 变量
❷   session_register("goodsid");                               //创建一个 session 变量
❸   session_register("goodsnum");                              //创建一个 session 变量
if($_SESSION["goodsid"]=="" && $_SESSION["goodsnum"]==""){   //判断 session 变量中的值是否为空
      $_SESSION["goodsid"]=$_GET["id"]."@";                    //如果为空则将商品的 ID 赋给变量
      $_SESSION["goodsnum"]="1@";                              //将商品数量设置为 1@
}else{
❹         $array=explode("@",$_SESSION["goodsid"]);           //如果不为空，则使用@分隔不同的商品 ID
      //判断如果获取的 ID 在 session 变量中已经存在，则提示该商品已经被放入购物车
❺   if(in_array($_GET["id"],$array)){
          echo "<script>alert('该编程词典已经被放入购物车！ ');history.back();</script>";
          exit;
      }
     $_SESSION["goodsid"].=$_GET["id"]."@";                    //为 session 变量赋值
     $_SESSION["goodsnum"].="1@";                              //为 session 变量赋值
}
//将商品放入购物车中，并跳转到购物车页
echo "<script>window.location.href='shopping_cart.php';</script>";
?>
```

说明

❶ session_start()：初始化 session 变量。

❷ session_register()：创建一个 session 变量 goodsid，存储商品 ID。

❸ session_register()：创建一个 session 变量 goodsnum，存储购买商品数量。

❹ explode()：将字符串依指定的字符串或字符进行分隔。返回由字符串组成的数组，每个元素都是 string 的一个子串，它们被字符串 separator 作为边界点分隔出来。如果设置了 limit 参数，则返回的数组包含最多 limit 个元素，而最后那个元素将包含 string 的剩余部分；如果 separator 为空字符串（""），explode()函数将返回 false；如果 separator 所包含的值在 string 中找不到，那么 explode()函数将返回包含 string 个元素的数组；如果参数 limit 是负数，则返回除了最后的-limit 个元素外的所有元素。

❺ in_array()：在指定的数组中搜索某个值，如果找到则返回 true，否则返回 false。

在实现购物车的分配和添加商品的功能后，接下来要做的就是查看购物车中的商品，即实现购物车中商品展示的功能。在购物车的商品展示中，可以实现清空购物车、删除购买商品、更改购买商品数量、继续购物和结算等操作。

购物车商品展示的功能主要通过 shopping_cart.php 文件来完成，首先从 session 变量中读取商品的 ID 和数量，然后根据商品的 ID 循环输出购物车中的商品，最后以商品 ID 为标识符设置不同的超链接，执行删除商品或者更改购买商品数量等操作。其代码如下：

（代码位置：光盘\TM\Instance\22\shopping_cart.php）

```
<table width="720" height="47" border="0" align="center" cellpadding="0" cellspacing="1" bgcolor="#999999">
<?php
    $array=explode("@",$_SESSION["goodsid"]);              //读取 session 变量中的商品 ID，以@进行分隔
    $arraynum=explode("@",$_SESSION["goodsnum"]);          //读取 session 变量中的商品数量，以@进行分隔
    $markid=0;                                             //创建变量，初始值为 0
    for($i=0;$i<count($array);$i++){                       //应用 for 循环语句循环输出商品 ID 的值
        if($array[$i]!=""){                                //判断如果商品 ID 的值不为空
            $markid++;                                     //增加变量$markid 的值
        }
    }
    if($markid==0){                                        //判断如果变量$markid 的值为空则输出下面的内容
?>
    <tr>
    <td height="22" colspan="4" bgcolor="#FFFFFF"><div align="center">对不起您的购物车中暂无商品信息!</div></td>
    </tr>
<?php
    }else{                                                 //如果$markid 的值不为空，则执行下面的内容
        $totalprice=0;                                     //创建变量$totalprice，初始值为 0
        for($i=0;$i<count($array);$i++){                   //循环输出数组中的商品 ID 值
            if($array[$i]!=""){
                //根据获取的商品 ID 的值，从数据库中获取对应产品的信息
                $sqlcart=mysql_query("select * from tb_bccd where id='".$array[$i]."'",$conn);
                $infocart=mysql_fetch_array($sqlcart)
?>
    <tr>
        <form name="form<?php echo $array[$i]?>" method="post" action="changegoodsnum.php">
      <td height="22" bgcolor="#FFFFFF"> <?php echo unhtml($infocart["bccdname"]);?></td>
      <td height="22" bgcolor="#FFFFFF"><div align="center"><?php echo
number_format($infocart["price"],2);?></div></td>
      <td height="22" bgcolor="#FFFFFF"><div align="center"><input type="text" name="goodsnum"
value="<?php echo $arraynum["$i"];?>" class="inputcss" size="8" ><input type="hidden" name="id"
value="<?php echo
$infocart["id"];?>" ></div></td>
      <td height="22" bgcolor="#FFFFFF"><div align="center"><a href="javascript:form<?php echo
$array[$i]?>.submit();" class="a1">更改数量</a> | <a href="delgoods.php?id=<?php echo $infocart["id"];?>"
```

```
class="a1">删除该项</a></div></td>
            </form>
       </tr>
<?php
                            $totalprice+=$infocart["price"]*$arraynum["$i"];
                      }
               }
       }
?>
</table>
```

22.6.4　商品订单的实现过程

在确定所要购买的商品后，接下来要做的就是进行购物结算，填写用户购物订单，将订单保存到数据库中，并且随机生成一个订单号，作为订单的唯一标识。生成订单的运行效果如图 22.27 所示。



图 22.27　生成订单的运行效果

提交订单后，用户可以选择汇款的方式：一是选择网上支付，那么将跳转到企业指定的网上银行进行汇款，汇款的操作将在企业指定的网上银行中进行，这里不做讲解；二是选择到指定的银行向企业提供的账号中汇款。企业将在收到汇款后按照用户指定的地址和方式配送产品。商品订单的生成和处理由 shopping_cart_getuserinfo.php 和 savebuyuser.php 文件来完成。

订单处理由 savebuyuser.php 文件完成，首先连接数据库，随机生成一个订单号，然后获取购物车中的商品信息，最后将商品信息和订单号存储到数据库中。其代码如下：

（代码位置：光盘\TM\Instance\22\savebuyuser.php）

```
<?php     session_start();                                                    //初始化 session 变量
include_once("conn/conn.php");                                               //连接数据库
$ddnumber=substr(date("YmdHis"),2,8).mt_rand(100000,999999);                 //随机生成订单号
$sql=mysql_query("select * from tb_city where id='".$_POST["city"]."'",$conn);  //读取数据库中的城市信息
$info=mysql_fetch_array($sql);
if($shfs=="1"){                                                              //判断用户选择的送货方式
      $yprice=$info['pt'];
      $shfs="普通邮递";
}elseif($shfs=="2"){
      $yprice=$info'[kd'];
      $shfs="邮政特快专递 EMS";
}
$array=explode("@",$_SESSION["goodsid"]);                    //以@来分隔 session 变量中存储的商品 ID
$arraynum=explode("@",$_SESSION["goodsnum"]);                //以@来分隔 session 变量中存储的商品数量
$totalprice=0;
for($i=0;$i<count($array);$i++){                             //循环读取数组中商品的 ID
```

```
        if($array[$i]!=""){
                $sqlcart=mysql_query("select * from tb_bccd where id='".$array[$i]."'",$conn);
                $infocart=mysql_fetch_array($sqlcart);
                $totalprice+=$infocart["price"]*$arraynum["$i"];
        }
}
$totalprice=$totalprice+$yprice;                                   //获取汇款金额
//将表单中提交的数据存储到数据库中
if(mysql_query("insert into
tb_dd(ddnumber,recuser,sex,address,yb,qq,email,mtel,gtel,shfs,spc,slc,yprice,totalprice,createtime,cityid)
values('".$ddnumber."','".$_POST["recuser"]."','".$_POST["sex"]."','".$_POST["address"]."','".$_POST["yb"]."','".$
_POST["qq"]."','".$_POST["email"]."','".$_POST["mtel"]."','".$_POST["gtel"]."','".$shfs."','".$_SESSION["goodsid"]
."','".$_SESSION["goodsnum"]."','".$yprice."','".$totalprice."','".date("Y-m-d H:i:s")."','".$_POST["city"]."')",$conn)){
        session_unregister("goodsid");                             //注销 session 变量 goodsid
        session_unregister("goodsnum");                            //注销 session 变量 goodsnum
        echo "<script>window.location.href='shopping_dd.php?ddno=".base64_encode($ddnumber)."';</script>";
}else{
        echo "<script>alert('订单信息保存失败，请重试!');</script>";
}
?>
```

22.7　社区论坛模块设计

视频讲解：光盘\TM\Video\第 22 章\社区论坛模块设计.exe

社区论坛模块为网站的浏览者提供一个交流的平台，以此来扩大网站的影响力，汇聚更多的人气，宣传企业形象，推广企业产品。

22.7.1　社区论坛模块概述

社区论坛模块为浏览者、会员、客户和企业提供了一个大的交流平台，根据身份的不同，给予不同的操作权限，社区论坛模块的操作流程如图 22.28 所示。

在本论坛中，浏览者只能够查看帖子；注册会员既可以查看帖子，也可以发布和回复帖子；管理员则具有发布、回复、查看和删除帖子的权限。

图 22.28　社区论坛操作流程图

22.7.2　页面跳转技术

在社区论坛模块的实现过程中，通过 JavaScript 脚本和下拉列表框的结合实现一个不同版块之间快速跳转的功能，从而能够更加灵活、方便地实现不同版块之间的跳转。

下面分析该技术是如何实现的。该技术的实现综合 3 个方面的内容，以一个下拉列表框为主，通过 PHP 语句从数据库中读取数据作为下拉列表框的值，应用 onchange 事件来调用 JavaScript 脚本，实现不同版块之间的跳转。这里以 bbs_top.php 文件中的快速跳转功能为例进行分析。其关键代码如下：

（代码位置：光盘\TM\Instance\22\bbs_top.php）

```
<!--创建一个下拉列表框，指定名称为 select_type，并且设置其属性，通过 onchange 事件来调用 JavaScript 脚本文件，实现页面跳转-->
<select name="select_type" class="inputcss"
❶    onChange="javascript:window.location=this.options[this.selectedIndex].value;" >
<?php
//通过 PHP 语句从数据库中读取数据，使用数据的 ID 作为下拉列表框的值，使用数据的标题 title 作为下拉列表框显示的内容
    $sql=mysql_query("select * from tb_type_small order by createtime desc",$conn);
    $info=mysql_fetch_array($sql);
    if($info==""){
        echo "<option>暂无讨论区</option>";
    }else{
        echo "<option>-版块快速跳转-</option>";
        do{                                  //应用 do…while 循环语句输出下拉列表框中的值
❷       echo "<option value='bbs_list.php?id=".$info['id']."'>".$info['title']."</option>";
        }
        while($info=mysql_fetch_array($sql));      //应用 do…while 循环语句结束
    }
?>
</select>
```

说明

❶ onChange：某元素失去焦点，并且从用户最后一次访问以来，其值已经改变。

location：用于访问窗口的当前定位（URL），既可被读取，又可被置换，可以通过其实现某个页面的定位或者更新。

❷ <option value='… '>…</option>：下拉列表框中输出的值，以及显示的内容。

该技术实现的效果如图 22.29 所示，它将实现从硬件咨询模块跳转到 PHP 模块。

22.7.3　论坛分类的实现过程

论坛分类实现两个功能，一是实现论坛中大的版块分区，分为 6 个版块：综合信息讨论区、操作系统、程序设计交流区、网管技术应用、Web 程序开发和数据库技术，其数据存储于 tb_type_big 数据表中。另一个是不同版块中不同语言和技术的分类，分为 11 种，其数据存储于 tb_type_small 表中。论坛分类的运行效果如图 22.30 所示。

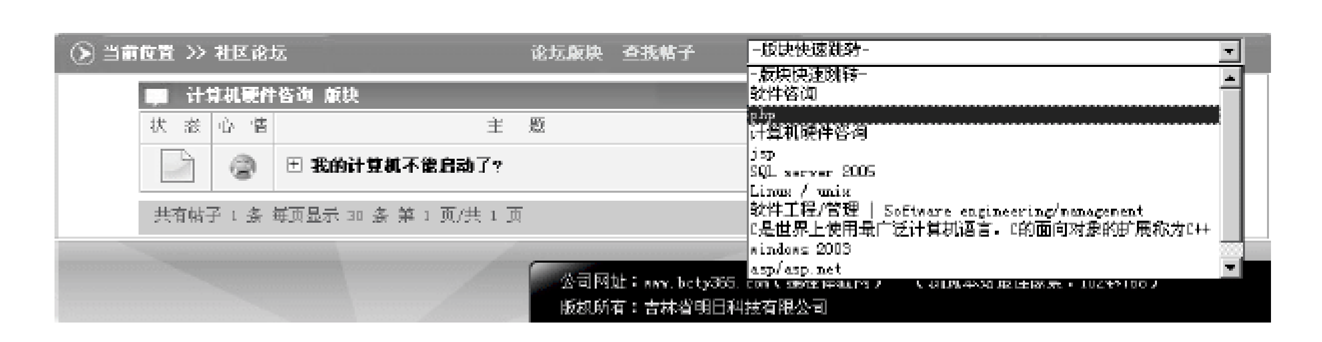


图 22.29　版块跳转功能的运行效果



图 22.30　论坛分类的运行效果



论坛分类的实现原理很简单，首先从 tb_type_big 表中读取 6 个版块中的数据，进行循环输出，然后在版块中嵌套循环，用于输出不用语言的分类数据。该功能主要通过 bbs_index.php 文件来完成。bbs_index.php 文件的程序代码如下：

（代码位置：光盘\TM\Instance\22\bbs_index.php）

```
<?php   include_once("top.php");                                    //调用网站头文件
        include_once("bbs_top.php");                                //调用社区论坛的头文件
?>
<?php
//循环输出数据表 tb_type_big 中的 6 个版块数据
$sql=mysql_query("select * from tb_type_big order by createtime desc",$conn);
$info=mysql_fetch_array($sql);
if($info==false){                                                   //如果返回值为 false，则执行下面的内容
?>
…                                                                   //省略了部分代码
<?php
    }else{                                                          //如果返回值为 true，则执行 do…while 循环语句
        /* 外部嵌套循环，输出论坛中的版块分类数据 */
❶       do{
?>
…                                                                   //省略了部分代码
<table  width="750"  border="0"  align="center"  cellpadding="0"  cellspacing="1"  bordercolor="#FFFFFF"
bgcolor="#6EBEC7">
<?php
    //循环输出 tb_type_small 表中的不同语言和技术的分类数据
    $sql1=mysql_query("select * from tb_type_small where bigtypeid='".$info["id"]."'",$conn);
    $info1=mysql_fetch_array($sql1);
    if($info1==false){                                              //判断如果返回值为 false，则执行下面的内容
?>
…                                                                   //省略了部分代码
<?php
     }else{    //如果返回值为 true，则执行下面的内容，输出该版块中对应语言和技术帖子的详细信息
?>
…                                                                   //省略了部分代码
❷   <?php    /* 内部嵌套循环，输出不同语言和技术的分类 */
             do{             ?>
        <tr>
            <td height="30"><font color="#666666">创建时间：<?php echo $info1["createtime"];?></font></td>
          </tr>
…                                                                   //省略了部分代码
      <?php
            }while($info1=mysql_fetch_array($sql1));
             /* 内部嵌套循环结束 */
     }
      ?>
</table>
  <?php
      }while($info=mysql_fetch_array($sql));
        /* 外部嵌套循环结束，对版块中的大类进行输出 */
}
 ?>
<?php   include_once("bottom.php");         ?>
```

说明

❶ do…while 循环语句，对论坛中大的版块分类进行循环输出。

❷ do…while 循环语句，对论坛中一个版块中的不同语言和技术进行循环输出。



在应用 do…while 循环语句时，while 后的分号不能省略。

22.7.4　论坛帖子浏览的实现过程

论坛帖子浏览主要输出指定帖子的详细信息，包括发帖人、用户级别、注册的时间，以及帖子的主题、内容和发帖时间，包括上传的图片。本模块中是用户权限使用体现的最明显的地方，可以分为 3 种情况：以浏览者进行登录，只能浏览帖子的内容，没有其他权限；以会员身份进行登录，可以对帖子进行回复，发表自己的看法；以管理员的身份进行登录，不但可以回复帖子，而且可以对任何人发布和回复的帖子进行删除和顶帖的操作。以管理员身份进行登录时的运行效果如图 22.31 所示。



图 22.31　管理员浏览帖子的结果图

论坛帖子浏览的功能通过 bbs_lookbbs.php 文件完成，首先根据传递的 ID 值读取指定的帖子数据，然后判断登录用户的类型，最后根据用户不同的类型执行不同的操作。其代码如下：

（代码位置：光盘\TM\Instance\22\bbs_lookbbs.php）

```
<?php
//根据$_GET 传递的数据获取 tb_bbs 中的数据
$sqlb=mysql_query("select * from tb_bbs where id='".$_GET["id"]."'",$conn);
$infob=mysql_fetch_array($sqlb);
//根据$_GET 传递的数据获取 tb_user 中的数据
$sql4=mysql_query("select * from tb_user where id='".$infob["userid"]."'",$conn);
$info4=mysql_fetch_array($sql4);
?>
…                              //省略了部分 HTML 代码
<table width="180" border="0" align="center" cellpadding="0" cellspacing="0">
        <tr>
        <td height="22">用户级别：
                        <?php
                                //根据用户信息表 tb_user 中字段 usertype 的值判断该用户的类型
                                //如果值为 1 则是管理员，值为 2 则是后台管理员，值为 0 则是普通会员
❶                              if($info4["usertype"]=="1") echo "管理员";else echo "普通会员";
                        ?>
                </td>
        </tr>
</table>
…                              //省略了部分 HTML 代码
<table width="500" height="200" border="0" align="center" cellpadding="0" cellspacing="0">
        <tr>
        <td width="35" height="20"><div align="center"><img src="images/lt_15(11).jpg" width="25"
height="25"></div></td>
            <td><?php echo $infob["createtime"];?></td>
        </tr>
        <tr>
        <td height="150" colspan="2">
❷                      <?php
                                //判断 tb_bbs 表中的字段 photo 是否为空，为空则执行下面的内容
                        if($infob['photo']!=""){
                                $photos=substr($infob['photo'],2,70);              //获取图片在服务器中的存储路径
```

```
                        echo (stripslashes($infob["content"]));        //输出帖子的内容
                      echo "<img src=\"$photos\">";                    //根据获取的图片路径，输出服务器中的图片
                 }else{                          //如果 tb_bbs 表中的图片字段 photo 为空，则执行下面的内容
❸                      echo (stripslashes($infob["content"]));        //只输出帖子的内容
                 }
               ?>
          </td>
     </tr>
</table>
<table width="530" height="20" border="0" align="center" cellpadding="0" cellspacing="0">
     <tr>
          <td width="239" height="30"> </td>
        <td width="291"><img src="images/lt_15(5).jpg" width="72" height="23" style="cursor:hand" onclick="
               <?php
                    //如果$_SESSION["unc"]的值为空，则不可以进行顶帖子的操作
                    if($_SESSION["unc"]==""){
                         echo "javascript:alert('请先登录本站，然后进行此操作！');window.location.href='index.php';";
                    }else{
                         //否则将判断当前用户的类型，如果是管理员则可以顶帖
                         $sqlu=mysql_query("select usertype from tb_user where usernc='".$_SESSION["unc"].
"'",$conn);
                         $infou=mysql_fetch_array($sqlu);
❹                        if($infou["usertype"]==1){                   //如果用户的类型为 1，则有顶帖的权限
                              echo "javascript:window.location.href='settop.php?id=".$infob["id"]."'";
                         }else{                                        //否则不具备该权限
                              echo "javascript:alert('对不起，您不具备该操作权限！');";
                         }
                    }
               ?>
               "/>  
❺              <?php
                    //判断当前用户是否具有删除帖子的权限
                    if($_SESSION["unc"]!=""){
                         //条件为用户不能为空，并且是管理员，才具备删除帖子的权限
                         $sqlu=mysql_query("select usertype from tb_user where
usernc='".$_SESSION["unc"]."'",$conn);
                    $infou=mysql_fetch_array($sqlu);
                         if($infou["usertype"]==1){
               ?>
               <img src="images/lt_15(10).jpg" onclick="javascript:if(window.confirm('您确定删除该帖吗？
')==true){window.location.href='bbs_delete.php?id=<?php echo $infob["id"]?>';}" style="cursor:hand"/>
               <?php
                         }
                    }
               ?>
          </td>
     </tr>
</table>
```

说明

❶ $info4["usertype"]：判断用户的类型，如果值为 1 是管理员，否则为普通会员。
❷ $infob['photo']：判断发布的帖子中是否含有图片，如果有则输出，没有则不输出。
❸ stripslashes()：将应用 addcslashes()函数处理后的字符串按原样返回。
❹ 判断登录用户是否具有顶帖的权限。
❺ 判断登录用户是否具有删除帖子的权限。

22.7.5　论坛帖子发布的实现过程

论坛帖子发布通过两个文件来完成，一个是帖子发布信息的提交页 bbs_pubs.php，另一个是对提交的数据进行处理的 retrieve.php 文件。帖子发布模块的运行效果如图 22.32 所示。

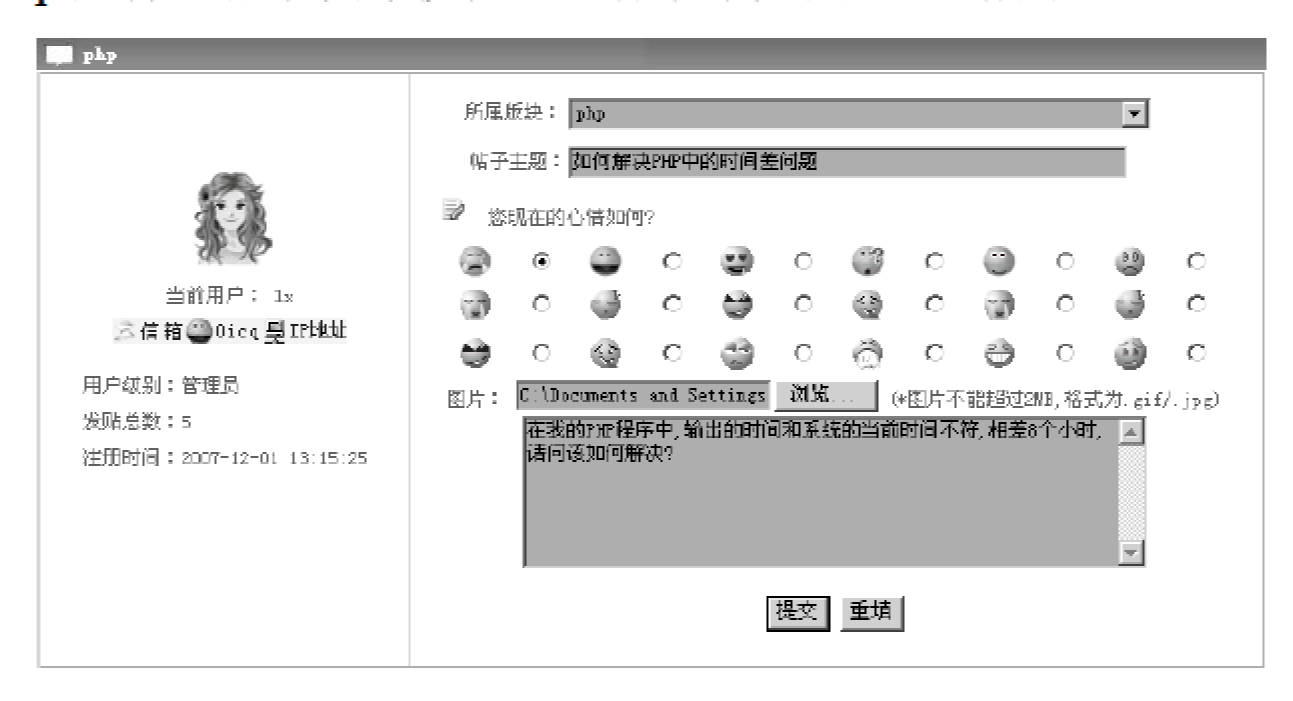


图 22.32　帖子发布模块的运行效果

在发布信息的提交页中，显示当前用户的个人信息，设置添加数据表单元素，其中表单元素的设计如表 22.3 所示。

表 22.3　发布信息页中使用的表单元素

名　称	元 素 类 型	重 要 属 性	含　义
form_bbs	form	method="post" action="retrieve.php" enctype="multipart/form-data"	发帖表单

bbs_type	select	class="inputcss" style="background-color:#6EBEC7">
		<?php
		$sql=mysql_query("select * from tb_type_small order by createtime desc",$conn);
		$info=mysql_fetch_array($sql);
		if($info==false){
		echo "<option>暂无讨论区</option>";
		}else{
		do{
		?>
		<option value="<?php echo $info['id'] ;?>"<?php if($_GET['id']==$info['id'])
		{echo "selected=\"selected\"";}?>><?php echo $info['title'];?></option>
		<?php }
		while($info=mysql_fetch_array($sql));
		}?>
			选择发表帖子的语言或者技术的类别

bbs_title	text	class="inputcss" style="background-color:#6EBEC7">	帖子标题
bbs_head	radio	value="<?php echo("images/bbsface/face".($i-1).".gif");?>"	表情图
bbs_photo	file	id="bbs_photo" class="inputcss" style="background-color:#6EBEC7" />	上传图片
content1	textarea	id="content1" class="inputcss" style="background-color:#6EBEC7">	帖子内容
Submit	submit	value="提交"	提交表单

在 retrieve.php 中对表单提交的数据进行处理，将数据存储到 tb_bbs 表中，并且更新用户信息表 tb_user 中 pubtimes 字段的值，其中还应用了图片上传技术，将图片上传到服务器中指定的文件夹下。retrieve.php 文件的代码如下：

（代码位置：光盘\TM\Instance\22\retrieve.php）



```
<?php      session_start();                                          //初始化 session 变量
$title=$_POST['bbs_title'];                                          //获取帖子的标题
$content=$_POST['content1'];                                         //获取帖子的内容
/* 判断提交的帖子主题和帖子内容是否为空*/
if($title==""){
        echo "<script>alert('请输入帖子主题!');history.back();</script>";
        exit; }
if($content==""){
        echo "<script>alert('请输入帖子内容!');history.back();</script>";
        exit; }
/*********/
include_once("conn/conn.php");                                       //连接数据库
//根据$_SESSION["unc"]的值读取数据库中用户的信息
$sql=mysql_query("select * from tb_user where usernc='".$_SESSION["unc"]."'",$conn);
$info=mysql_fetch_array($sql);                                       //检索指定条件的数据信息
$userid=$info['id'];                                                 //获取用户 id
$typeid=$_POST['bbs_type'];                                          //接收版块名称
$title=$_POST['bbs_title'];                                          //接收帖子主题
$content=$_POST['content1'];                                         //接收帖子内容
$head=$_POST['bbs_head'];                                            //接收头像
$createtime=date("Y-m-j H:i:s");                                     //获取系统当前时间
$lastreplytime=$createtime;                                          //将当前时间赋给变量
$readtimes=0;
$link=date("YmjHis");
❶      if($_FILES['bbs_photo']["name"]==true){  //上传图片，判断文件是否存在，如果存在则执行下面的内容
❷      $photo_name=strtolower(stristr($_FILES["bbs_photo"]["name"],"."));   //获取图片后缀名，将字符转成小写
        if($photo_name!=".gif" & $photo_name!=".jpg" & $photo_name!=".jpeg" ){ //判断图片的格式是否符合要求
        echo "<script>alert('您上传的图片格式不正确!');history.back();</script>";
        }else{
❸              $paths1=$link.mt_rand(1000000,9999999).$photo_name;        //创建图片的名称
        $photos="./upfile/".$paths1;                                      //创建图片的存储路径
❹                  move_uploaded_file($_FILES['bbs_photo']["tmp_name"],$photos);   //将图片存储到指定的文件夹下
                //向数据库添加数据
                if(mysql_query("insert into tb_bbs(userid,typeid,title,content,createtime,lastreplytime,head,readtimes,
top,photo)
values('".$userid."','".$typeid."','".$title."','".$content."','".$createtime."','".$lastreplytime."','".$head."','".$readtimes
."','0','$photos')",$conn)){
                        mysql_query("update tb_user set pubtimes=pubtimes+1",$conn);   //更新 tb_user 中 pubtimes 字段的值
                        echo "<script>alert('新帖发表成功!');history.back();</script>";
                }else{
                        echo "<script>alert('新帖发表失败!');history.back();</script>";
                }
        }
}else{                  //如果没有提交图片，则执行下面的内容
        if(mysql_query("insert  into  tb_bbs(userid,typeid,title,content,createtime,lastreplytime,head,readtimes,top)
values
('".$userid."','".$typeid."','".$title."','".$content."','".$createtime."','".$lastreplytime."','".$head."','".$readtimes."','0')",
$conn)){
                mysql_query("update tb_user set pubtimes=pubtimes+1",$conn);
                echo "<script>alert('新帖发表成功!');history.back();</script>";
        }else{
                echo "<script>alert('新帖发表失败!');history.back();</script>";
        }
}
?>
```

说明 ❶ $_FILES['bbs_photo']["name"]：$_FILES[]全局变量，获取表单提交文件的原始名称。

❷ strtolower()：将指定的字符转换为小写字母。

stristr()：获取指定字符串（A）在另一个字符串（B）中首次出现的位置到（B）字符串末尾的所有字符串。该函数如果执行成功则返回剩余的字符串，否则将返回 false。

❸ mt_rand()：生成一个随机数，用于上传文件的名称。

❹ move_uploaded_file()：将指定的文件上传到指定的文件夹下。

22.7.6　论坛帖子回复的实现过程

回复论坛中的帖子，必须是以会员或者管理员的身份进行登录，否则不能进行帖子的回复操作，其运行效果如图 22.33 所示。

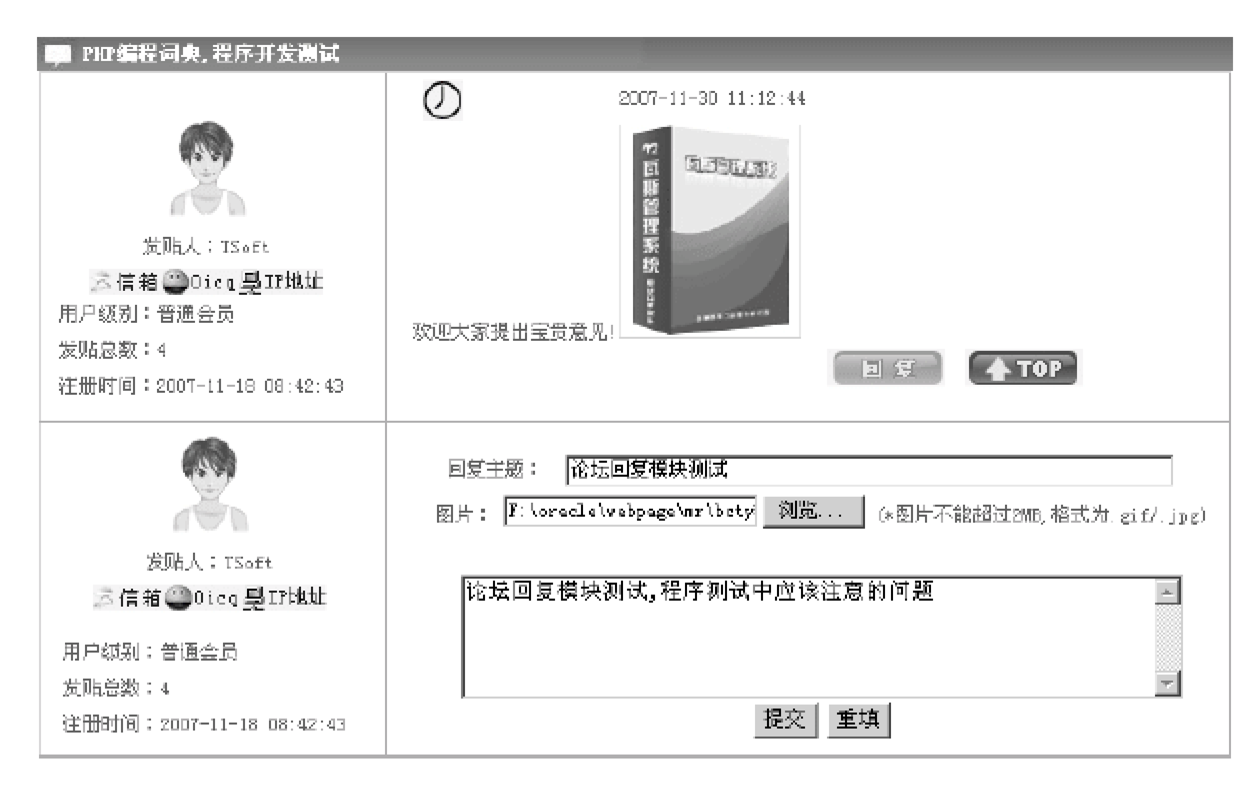


图 22.33　论坛帖子回复的运行效果

论坛帖子回复功能主要通过 bbs_looks.php 和 savereply.php 文件实现。其中应用 JavaScript 脚本对回复帖子的文本框进行输出和隐藏的控制。在 bbs_looks.php 文件中，帖子回复使用的表单元素如表 22.4 所示。

表 22.4　论坛帖子回复中的重要表单元素

名　　称	元 素 类 型	重 要 属 性	含　　义
form_reply	form	method="post" action="savereply.php" enctype="multipart/form-data">	回复表单
reply_title	text	class="inputcss" id="reply_title"	回复帖子主题
bbsid	hidden	value="<?php echo $infob["id"];?>"	对应帖子的 ID
bbs_head	radio	value="<?php echo("images/bbsface/face".($i-1).".gif");?>"	表情图
bbs_photo	file	id="bbs_photo" class="inputcss"	上传图片
content1	textarea	id="content1"	回复帖子内容
Submit	submit	value="提交"	提交表单

在帖子回复表单 bbs_looks.php 页中，首先判断登录用户是否具有回复的权限，然后根据提交的值展开回复表单的文本框，在文本框中输入回复的主题和内容，最后将数据提交到表单处理页 savereply.php 中。bbs_looks.php 的主要代码如下：

（代码位置：光盘\TM\Instance\22\bbs_lookbbs.php）

```
<script language="javascript">
//设计回复帖子表格的输出方式
function show_reply(){                                              //定义一个函数
	if(reply_bbs1.style.display=="")                                //判断当 display 的值为空时
		{
			reply_bbs1.style.display="none" ;                       //则输出表格
			button_show_bbs.value="回复帖子";                       //显示“回复帖子”
		}
	else if(reply_bbs1.style.display=="none")                       //判断当 display 的值为 none 时
		{
			reply_bbs1.style.display="" ;                           //则不输出表格
			button_show_bbs.value="关闭窗口";                       //显示“关闭窗口”
		 }
}
</script>
<!-- --------------------------------------判断登录用户是否具有回复的权限----------------------------------------- -->
<img src="images/lt_15(9).jpg" width="72" height="23" id="button_show_bbs"  style="cursor:hand" onClick="
<?php
	if($_SESSION["unc"]==""){
		echo "javascript:alert('请先登录本站，然后回复帖子!');window.location.href='index.php';";
	}else{
?>
	show_reply()
<?php   }     ?>        "/>
```

表单处理页 savereply.php 将表单提交的数据存储到指定的数据库中，其实现的方法与论坛发布中的表单处理技术是相同的，有关该技术的详细讲解请参考 22.7.5 节，这里不再赘述。

22.8　后台首页设计

视频讲解：光盘\TM\Video\第 22 章\后台首页设计.exe

作为一个完整的网上社区系统，要想能够及时地对网站进行管理和维护，必须具有一个强大的后台管理系统，对网上社区系统中的数据进行更新和维护。

22.8.1　后台首页概述

网上社区系统的后台管理采用的是一种简单的框架结构，通过 switch 语句来实现。其具体内容如下。

- ☑ 软件试用管理：包括软件试用产品的添加和删除。
- ☑ 编程词典管理：包括编程词典版本和内容的添加、删除。
- ☑ 在线订购管理：主要用于管理用户提交的订单。
- ☑ 软件升级管理：包括升级包和序列号的添加、删除。
- ☑ 站内公告管理：主要用于添加和删除站内公告。
- ☑ 技术支持管理：主要用于添加和删除常见问题，以及对客户反馈信息进行管理。

本项目中提供的后台首页在本书光盘中的路径为\TM\22\admin\index.php，打开效果如图 22.34 所示。

22.8.2　switch 框架技术

网上社区后台首页的设计主要应用 switch 语句和 include 包含语句，其实现的原理是：应用 switch 语句，

根据超链接中传递的变量值进行判断，根据不同的变量值应用 include 调用不同的子文件。该技术的实现流程如图 22.35 所示。



图 22.34　BCTY365 网上社区系统后台首页



图 22.35　网上社区后台首页设计流程

为了更好地理解 switch 框架技术，先来了解一下 switch 语句。该语句的格式如下：

```
switch( expr ){                                    //expr 条件为变量名称
     case expr1:                                   //case 后的 expr1 为变量的值
          statement1;                              //冒号（:）后是符合该条件时要执行的部分
     break ;                                       //应用 break 来跳离循环体
    case expr2 :
         statement2 ;
     break ;
     default:
          statementN;
     break;
}
```

参数 expr 是表达式的值，即 switch 语句的条件变量的名称；参数 expr1 放置于 case 语句后，是要与条件变量 expr 进行匹配的值中的一个；statement1 是在参数 expr1 的值与条件变量 expr 的值相匹配时执行的代码；break 语句终止语句的执行，即当语句在执行过程中，遇到 break 就停止执行，跳出循环体；default 是 case 的一个特例，匹配了任何其他 case 都不匹配的情况，并且是最后一条 case 语句。

通过 switch 和 include 语句来实现后台管理功能的设计是一个很好的方法，不但实现过程简单，而且操作也非常灵活。其关键代码如下：

（代码位置：光盘\TM\Instance\22\admin\wzdh.php）

```
<?php
switch($htgl){                                     //根据变量提交的不同值
   case "添加编程词典版本":                          //判断与变量提交的值是否相同
       include("addbb.php");                       //如果值相同，则调用指定的文件
   break;                                          //跳出本次循环
  case "编辑编程词典版本":
  include("editbd.php");
   break;
     …                                             //部分代码省略
     case "":                                      //当变量的值为空时
  include("edittell.php");                         //调用该文件
   break;
}
?>
```

22.8.3 后台首页的实现过程

在后台首页的设计过程中，以 switch 循环语句为基础，架设整个后台管理功能的框架结构；充分发挥 include 包含语句的作用，调用不同的文件执行不同的管理操作；应用 JavaScript 脚本来控制栏目列表的输出和隐藏。

控制栏目列表的输出和隐藏在 menu.php 文件中进行，首先定义一个 change()函数用于控制表格的输出和隐藏，然后在表格中应用 onclick 事件传递不同的值到自定义函数 change()，最后根据不同的值显示不同的内容。其关键代码如下：

（代码位置：光盘\TM\Instance\22\admin\menu.php）

```
<script language="javascript">
//通过脚本语言控制文本框的伸展和收缩
function change(x,y){                                          //定义一个函数
        if(x.style.display=="none"){                           //判断当样式的值为 none 时
             x.style.display="";                               //输出样式的值为空
        }
        else if(x.style.display==""){                          //判断当样式的值为空时
             x.style.display="none";                           //输出样式的值为 none
             y.background="images/bg_16_11.jpg";               //输出背景图片
               }
        }
</script>
<table width="175" height="26" border="0" align="center" cellpadding="0" cellspacing="4"
onclick="change(tz1,img_tz1)" style="cursor:hand">
   <tr>
      <td background="images/bg_16_11.jpg" id="img_tz1" class="a4"><div align="left"><img
src="images/bg_16_21.jpg"> 编程词典管理</div></td>
   </tr>
</table>
<table name="tz1" id="tz1" width="170" height="40" border="0" align="center" cellpadding="0" cellspacing="0"
<?php
//根据变量的值选择执行的内容，当变量的值不为真时，隐藏该表格
if(isset($_GET['htgl'])){
        if($_GET['htgl']!="添加编程词典版本" || $_GET['htgl']!="编辑编程词典版本" ||$_GET['htgl']!="添加编程词典"
||$_GET['htgl']!="编辑编程词典" ){
?>
                          style="display:none"
                          <?php
                          }
                           }
                          ?>
>
     <tr>
      <td width="40" height="24" background="images/bg_16_16.jpg"> </td>
      <td width="114" background="images/bg_16_16.jpg"><div align="left"><a href="default.php?htgl=添加编程
词典版本">添加编程词典版本</a></div></td>
   </tr>
</table>
```

说明 这里给出的只是后台首页实现过程中的主要代码，详细代码可参考本书光盘 TM\22\admin 文件夹下的相关文件。

22.9 编程词典管理模块设计

视频讲解：光盘\TM\Video\第 22 章\编程词典管理模块设计.exe

本模块的功能是对网站中的编程词典进行管理，包括添加编程词典版本、编辑编程词典的版本、添加编程词典和编辑编程词典。

22.9.1 编程词典管理模块概述

本模块的主要功能是管理网站中在线出售的编程词典软件，实现对编程词典软件的更新和维护，其管理的内容主要包括添加和编辑编程词典的版本、添加和编辑编程词典的详细信息。在添加编程词典时，包括名称、版权、图片、类别、内容简介和不同版本的共同点；编辑编程词典包括版本、价格、简介、功能和服务，其中每一个编程词典软件只可以编辑一次，不可以重复编辑，如果要重新编辑，就必须将已经编辑过的信息删除。

22.9.2 图片上传技术

在编程词典管理模块中，应用到图片上传技术，通过该技术将编程词典的界面效果上传到服务器的指定文件夹下。该技术主要通过 move_uploaded_file()函数来实现，其中还应用到 is_dir()、mkdir()函数，判断指定的文件夹是否存在和创建文件夹，还有 mt_rand()、strstr()函数和$_FILES[]全局变量。为了更好地理解和掌握图片上传处理技术，这里以编程词典模块中的 savebccd.php 文件为例进行讲解。

首先应用 is_dir()函数判断在服务器中是否存在指定的文件夹，如果不存在则应用 mkdir()函数创建一个新的文件夹。

然后应用$_FILES[]全局变量获取图片名，应用 strstr()函数获取图片文件的后缀名，为避免出现同名文件覆盖，这里应用系统的当前时间和 mtrand()函数获取的一个 7 位随机数字作为图片的名称。

最后确定图片在服务器中存储的路径，将图片上传到指定的文件夹下。而数据库中存储的数据是图片在服务器中的路径，当需要输出图片时，只需要获取数据库中图片的路径即可。savebccd.php 文件的代码如下：

（代码位置：光盘\TM\Instance\22\admin\savebccd.php）

```
<?php include_once("../conn/conn.php");                          //连接数据库
$bccdname=$_POST['bccdname'];                                     //获取 POST 方法提交的值
$owner=$_POST['owner'];
$typeid=$_POST['typeid'];
$content=$_POST['content'];
$samepart=$_POST['samepart'];
$addtime=date("Y-m-j H:i:s");                                     //获取当前时间
❶    if(is_dir("./bccdimages")==false){                          //判断指定的文件是否存在
❷    mkdir("./bccdimages");                                       //如果不存在，则创建一个新的文件夹
}
$link=date("YmjHis");                                             //获取当前时间
//为表单中提交的数据重新命名，以当前时间和随机数作为名称
//其中使用$_FILES 获取表单中真实的名称，使用 strstr 函数获取文件的后缀
❸    $path=$link.mt_rand(1000000,9999999).strstr($_FILES["imageaddress"]["name"],".");
$address="./bccdimages/".$path;                                    //定义文件上传的路径
❹    move_uploaded_file($_FILES["imageaddress"]["tmp_name"],$address);//将文件上传到指定的文件中
```

```
$imageaddress="./admin/bccdimages/".$path;                    //获取上传文件在服务器中的存储路径
//将表单中提交的数据存储到数据库中
$query=mysql_query("insert into tb_bccd(bccdname,owner,typeid,content,samepart,imageaddress,addtime)
values('$bccdname','$owner','$typeid','$content','$samepart','$imageaddress','$addtime')",$conn);
if($query==true){
    echo "<script>alert('编程词典添加成功！');history.back();</script>";
}else{
    echo "<script>alert('编程词典添加失败！');history.back();</script>";
}
?>
```

说明

❶ is_dir()：判断指定的文件夹是否存在，如果存在则返回 true，否则返回 false。

❷ mkdir()：创建一个新的文件夹。

❸ mt_rand()：根据提供的参数 min 和 max 生成随机数，如果没有提供可选参数 min 和 max，则返回 0 ~ RAND_MAX 之间的伪随机数。

strstr()：获取一个指定字符串在另一个字符串中首次出现的位置到后者末尾的子字符串。如果执行成功，则返回剩余字符串（存在相匹配的字符）；如果没有找到相匹配的字符，则返回 false。

$_FILES[]：全局变量，获取所有上传文件的信息。该全局变量还可以获取到其他的值，其中 $_FILES['imageaddress']['name']获取的是客户端机器文件的原名称；$_FILES['imageaddress']['size']获取已上传文件的大小，单位为字节；$_FILES['imageaddress']['tmp_name']获取文件被上传后在服务端储存的临时文件名；$_FILES['imageaddress']['error']获取和该文件上传相关的错误代码。

❹ move_uploaded_file()：应用 POST 方法实现文件的上传，参数 filename 指定要上传的文件地址；参数 destination 指定文件上传到服务器后的存储目录及名称。

注意

应用 POST 方法上传图片文件时，应当在上传表单的 form 标记中添加以下内容：enctype="multipart/form-data"。

22.9.3　添加编程词典模块的实现过程

添加编程词典模块的功能是向数据库中添加编程词典的详细信息，包括编程词典的名称、版权、图片、类别、内容简介和不同版本的共同点。其运行效果如图 22.36 所示。

添加编程词典	
编程词典名称：	PHP编程词典
版权所有：	吉林省明日科技有限公司
图片路径：	C:\Documents and Settings\ 浏览...
所属类别：	『 PHP 』类
内容简介：	整合当前大部分比较流行的PHP以及相关的技术，为广大的PHP爱好者提供一个好助手.
不同版本相同点：	针对不同的版本，其中涉及到不同难易程度的知识
	添加　重写

图 22.36　添加编程词典模块的运行效果

添加编程词典模块主要通过 addbccd.php 和 savebccd.php 文件来完成，其中 addbccd.php 文件主要用于

设计表单元素，而 savebccd.php 文件主要是对表单中提交的数据进行处理。addbccd.php 文件中使用的表单元素如表 22.5 所示。

表 22.5　添加编程词典页中使用的重要表单元素

名　　称	元 素 类 型	重 要 属 性	含　　义
form1	form	method="post"　action="savebccd.php"　onSubmit="return　chkinput(this)" enctype="multipart/form-data">	编程词典表单
bccdname	text	size="25" class="txt_grey"	编程词典名称
owner	text	" size="25" class="txt_grey"	版权所有者
imageaddress	file	size="25" class="txt_grey"	界面图片

typeid	select	<?php　include_once("../conn/conn.php");
		$sql=mysql_query("select * from tb_type order by createtime desc",$conn);
		$info=mysql_fetch_array($sql);
		if($info==false){ echo "<option >暂无类别</option>";
		}else{
		do{
		　echo "<option value=".$info['id'].">".$info['typename']."</option>"; }
		while($info=mysql_fetch_array($sql));}　　　?>
			选择编程词典的版本

content	textarea	rows="10" cols="65" class="textarea">	内容简介
samepart	textarea	rows="10" cols="65" class="textarea">	不同版本的 特点
Submit	submit	value="添加" class="btn_grey"	提交表单

savebccd.php 文件实现对表单中提交的数据进行处理，首先通过$_POST 获取表单中提交的数据，然后判断指定的文件夹是否存在，最后将数据存储到指定的数据表中。其关键代码如下：

（代码位置：光盘\TM\Instance\22\admin\savebccd.php）

```
<?php include_once("../conn/conn.php");                          //连接数据库
$bccdname=$_POST['bccdname'];                                    //获取 POST 方法提交的值
$owner=$_POST['owner'];
$typeid=$_POST['typeid'];
$content=$_POST['content'];
$samepart=$_POST['samepart'];
$addtime=date("Y-m-j H:i:s");                                    //获取当前时间
❶    if(is_dir("./bccdimages")==false){                          //判断指定的文件是否存在
❷    mkdir("./bccdimages");                                      //如果不存在，则创建一个新的文件夹
}
$link=date("YmjHis");                                            //获取当前时间
//为表单中提交的数据重新命名，以当前时间和随机数作为名称，其中使用$_FILES 获取表单中真实的名称，使用
strstr()函数获取文件的后缀
❸    $path=$link.mt_rand(1000000,9999999).strstr($_FILES["imageaddress"]["name"],".");
$address="./bccdimages/".$path;                                        //定义文件上传的路径
move_uploaded_file($_FILES["imageaddress"]["tmp_name"],$address);      //将文件上传到指定的文件中
$imageaddress="./admin/bccdimages/".$path;                             //获取上传文件在服务器中的存储路径
//将表单中提交的数据存储到数据库中
$query=mysql_query("insert into tb_bccd(bccdname,owner,typeid,content,samepart,imageaddress,addtime)
values('$bccdname','$owner','$typeid','$content','$samepart','$imageaddress','$addtime')",$conn);
if($query==true){
      echo "<script>alert('编程词典添加成功!');history.back();</script>";
}else{
      echo "<script>alert('编程词典添加失败!');history.back();</script>";
}
?>
```



说明

❶ is_dir()：判断指定的文件是否存在。

❷ mkdir()：创建一个新的文件夹。

❸ mt_rand()：获取随机数字。

strstr()：获取一个指定字符串在另一个字符串中首次出现的位置到后者末尾的子字符串。如果执行成功，则返回剩余字符串（存在相匹配的字符）；如果没有找到相匹配的字符，则返回 false。

22.9.4 编辑编程词典模块的实现过程

在完成对编程词典信息的添加后，接下来就可以对编程词典的版本信息进行编辑，主要添加版本信息、价格、简介、功能和推出的服务。该模块的运行效果如图 22.37 所示。

图 22.37 编辑编程词典模块的运行效果

该功能的实现同样通过两个文件，一个是提交表单的 editbccd.php 文件，一个是处理表单提交数据的 sacvbccdbb.php 文件。提交表单文件 editbccd.php 中使用的表单元素如表 22.6 所示。

表 22.6 编辑编程词典页中使用的重要表单元素

名　称	元素类型	重要属性	含　义
form1	form	method="post" action="savebccdbb.php" onSubmit="return chkinput(this)">	编辑编程词典表单
bccdname	text	class="txt_grey" disabled="disabled" value=" <?php $sql4=mysql_query("select bccdname from tb_bccd where id='".$_GET['bccdid']."'",$conn); $info4=mysql_fetch_array($sql4); echo unhtml($info4['bccdname']); ?>	编程词典名称，这里设置了文本的只读属性
bccdid	hidden	value="<?php echo $_GET['bccdid'];?>	编程词典的 ID

续表

名　称	元素类型	重要属性	含　义
bbid	select	<?php $sql3=mysql_query("select * from tb_bb order by createtime desc ",$conn); $info3=mysql_fetch_array($sql3); if($info3==false){ echo "<option>暂无版本信息</option>"; }else{ do{ ?> <option value="<?php echo $info3['id'];?>"><?php echo unhtml($info3['bbname']);?></option> <?php }while($info3=mysql_fetch_array($sql3)); }?>	选择编程词典的版本
Submit	submit	value="添加" class="btn_grey"	提交表单

sacvbccdbb.php 文件对表单提交的数据进行处理，首先获取表单中提交的数据，然后判断指定的版本是否已经被添加，最后将数据存储到指定的数据表中。其代码如下：

（代码位置：光盘\TM\Instance\22\admin\savebccdbb.php）

```
<?php
$bccdid=$_POST['bccdid'];                                    //获取表单中提交的数据
$bbid=$_POST['bbid'];                                        //获取编程词典 id
$price=$_POST['price'];                                      //获取编程词典单价
$content=$_POST['content'];                                  //获取编程词典内容
$gn=$_POST['gn'];                                            //获取编程词典功能
$fw=$_POST['fw'];                                            //获取编程词典服务
include_once("../conn/conn.php");                            //连接数据库文件
//判断提交的编程词典是否已经被添加
$sql=mysql_query("select id from tb_bbqb where bccdid='".$bccdid."'",$conn);
$info=mysql_fetch_array($sql);                               //检索指定编程词典的 id
if($info!=false){                                            //如果检索值为真，则弹出提示
     echo "<script>alert('该版编程词典已经添加!');history.back();</script>";
     exit; }
/* ****************************************************************** */
$query=mysql_query("insert into tb_bbqb(bccdid,bbid,price,content,gn,fw)
values('$bccdid','$bbid','$price','$content','$gn','$fw')",$conn);    //将表单中提交的数据存储到数据库中
//更新编程词典的价格
$querys=mysql_query("update tb_bccd set bbid='$bbid',price='$price' where id='".$bccdid."'");
if($query==true and $querys==true){                          //如果添加和更新操作为真，则弹出提示
     echo "<script>alert('版本信息添加成功!');history.back();</script>";
}else{                                                       //如果添加和更新操作为假，则弹出提示
     echo "<script>alert('版本信息添加失败!');history.back();</script>";
}
?>
```

22.10　软件升级管理模块设计

视频讲解：光盘\TM\Video\第 22 章\软件升级管理模块设计.exe

22.10.1　软件升级管理模块概述

软件升级管理模块实现对软件升级包的管理，其具体的功能包括添加升级包、编辑升级包、添加序列

号和编辑序列号。软件升级管理模块中的添加升级包和添加序列号是一一对应的，其中根据所属的类别和版本来确定升级包对应的序列号，每一个版本一个类别的升级包对应一个序列号。

22.10.2　动态输出下拉列表框的值

在软件升级管理模块中，应用到一个动态输出下拉列表框中值的技术。下面就来讲解一下该技术是如何实现的。

在讲解该技术之前，先来了解一下下拉列表框的基本结构。

```
<select name="select"><!-->name 指定该下拉列表框的名称<!-->
	<!-->selected 设置下拉列表框的默认值，默认值为 PHP<!-->
	<option selected="selected">PHP</option>
	<!-->value 指定的"mysql"是下拉列表框传递的值，"MYSQL"为显示的内容<!-->
	<option value="mysql">MYSQL</option>
</select>
```

所谓动态输出下拉列表框中的值就是从数据库中读取数据，将获取到的数据输出到下拉列表框中，而不是直接在下拉列表框中设置某个固定的值。这里以软件升级管理模块文件 addsjb.php 中的所属类别下拉列表框为例进行讲解，其中设置下拉列表框的名称为 typeid，默认值为“请选择”，value 的值是从数据库中获取的 ID 值，显示的内容为从数据库中获取的类型名称。动态输出下拉列表框中值使用的关键代码如下：

（代码位置：光盘\TM\Instance\22\admin\wzdh.php）

```
<!-- ----------------------------------------设置下拉列表框的名称为 typeid---------------------------------------- -->
<select name="typeid" class="txt_grey">
	<option value="" selected="selected">请选择</option>
	<?php
		include_once("../conn/conn.php");				//连接数据库
		//从数据库中读取编程词典类型的数据
		$sql=mysql_query("select * from tb_type order by createtime desc",$conn);
		$info=mysql_fetch_array($sql);
		if($info==false){
			echo "<option >暂无类别</option>";
		}else{
			do{							//应用 do…while 循环语句输出类型的 ID 和名称
				echo "<option value=".$info['id'].">".$info['typename']."</option>";
			}
			while($info=mysql_fetch_array($sql));		//do…while 循环语句结束
		}
	?>
</select>
```

下拉列表框不但可以动态输出数据库中某个字段的数据，而且还可以输出数组中的数据。下面就实现一个在下拉列表框中动态输出数组中数据的功能，首先创建一个下拉列表框，然后设置下拉列表框的值，从数组中读取数据，应用 for 循环语句进行输出。其代码如下：

```
<!-- -----------------------------------设置下拉列表框的名称为 select------------------------------------- -->
<select name="select" size="1">
<?php
	$string="ASP,PHP,JSP,.NET,DEL,VB,VC";			//定义一个字符串
	$srtings=split(",",$string);				//分隔字符串
	$count=count($srtings);					//获取数组中元素的数量
	for($i=0;$i<$count;$i++){				//根据数组中元素的数量进行循环输出
		$result=$srtings[$i];				//定义变量，获取数组中指定的元素
	echo "<option>$result</option>";}			//将数组中的元素输出到下拉列表框中
```

```
?>
</select>
```

动态输出数据库中数据到下拉列表框的运行效果如图 22.38 所示。

22.10.3 软件升级包上传的实现过程

软件升级包上传在添加升级包模块中实现，通过一个文件域文本框将升级包提交到服务器中指定的文件下，并且将该文件在服务器中的路径存储到数据库中，便于在前台实现对软件升级包的下载。其运行效果如图 22.39 所示。

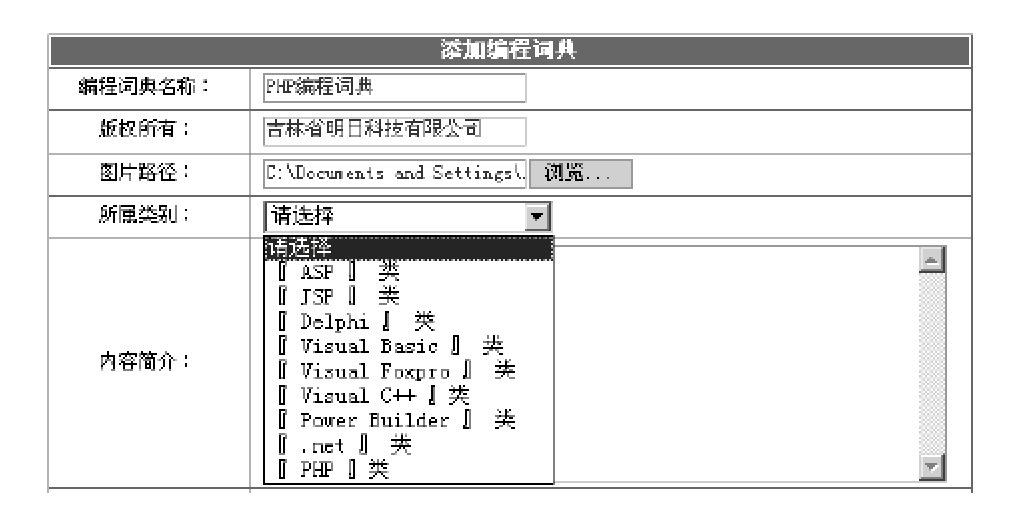


图 22.38 动态输出数据库中数据到下拉列表框



图 22.39 软件升级包上传的运行效果

在本模块中通过 addsjb.php 文件来提交升级包的信息，通过 savesj.php 文件来对表单提交的数据进行处理。其中在将升级包上传到服务器的指定文件夹的过程中，主要应用的是 move_uploaded_file()函数。在 savesj.php 文件中，首先获取表单提交的数据，然后判断服务器中是否存在指定的文件，最后应用 move_uploaded_file()函数将升级包上传到指定的文件夹下，并且将数据存储到指定的数据表中。其程序代码如下：

（代码位置：光盘\TM\Instance\22\admin\savesj.php）

```
<?php
$name=$_POST['name'];                                    //获取表单提交的数据
$typeid=$_POST['typeid'];                                //获取表单提交的数据
$content=$_POST['content'];                              //获取表单提交的数据
$addtime=date("Y-m-j H:i:s");                            //定义时间变量
$bbid=$_POST['bbid'];                                    //获取表单提交的数据
if(is_dir("./sjxz")==false){                             //判断指定的文件夹是否存在
    mkdir("./sjxz");                                     //如果指定的文件夹不存在，则创建一个指定的文件夹
}
$link=date("YmjHis");                                    //获取一个时间
$path=$link.mt_rand(1000000,9999999).strstr($_FILES["address"]["name"],".");  //重新设置升级包名称
$address="./sjxz/".$path;                                //设置升级包在服务器中存储的指定路径
move_uploaded_file($_FILES["address"]["tmp_name"],$address);          //将升级包上传到指定的路径下
$address="./admin/sjxz/".$path;                          //获取升级包在服务器中的存储路径
include_once("../conn/conn.php");                        //连接数据库文件
//将上传的数据存储到数据库中，这里将升级包在服务器中的路径存储到数据库中
$query=mysql_query("insert into tb_sjxz(name,typeid,content,addtime,address,bbid)
values('$name','$typeid','$content','$addtime','$address','$bbid')",$conn);
if($query){                                              //如果添加操作成功，则弹出提示
    echo "<script>alert('升级包添加成功!');history.back();</script>";
}else{                                                   //如果添加操作失败，则弹出提示
    echo "<script>alert('升级包添加失败!');history.back();</script>";
}
?>
```

22.10.4 软件升级包删除的实现过程

软件升级包删除的实现主要根据当前数据中提供的 ID，执行 delete 删除语句，将数据表中相同 ID 的数据删除。其运行效果如图 22.40 所示。



图 22.40 软件升级包删除的运行效果

该功能的实现主要通过 editsjb.php 和 deletesjb.php 文件。通过 editsjb.php 文件输出数据库中存储的有关升级包的信息，以分页的形式显示，在每条记录的最后设置一个删除链接，通过脚本来调用 deletesjb.php 文件，根据变量中的 ID 值执行删除升级包的操作。其关键代码如下：

（代码位置：光盘\TM\Instance\22\admin\deletesjb.php）

```
<?php
$id=$_GET['id'];                                                    //获取变量传递的 ID
include_once("../conn/conn.php");                                   //连接数据库
//执行删除操作，将数据表中对应的 ID 的数据删除
if(mysql_query("delete from tb_sjxz where id='".$id."'",$conn)){
	echo "<script>alert('该升级包删除成功!');history.back();</script>";	//如果删除操作成功，则弹出提示
}else{                                                              //如果删除操作失败，则弹出提示
	echo "<script>alert('该升级包删除失败!');history.back();</script>";
}
?>
```

22.11 在线支付技术专题

所谓在线支付就是客户端（金融机构需客户端安装由金融机构签发的数字证书，信用卡免安装）将支付信息加密后通过互联网传送到支付网关（支付网关是解决网络上安全支付问题的交易平台，位于互联网和传统的金融机构内部网之间，其主要作用是将互联网和金融网络安全地连接起来，将不安全的网上交易信息传给安全的金融网络，起到隔离和保护金融网络的作用），同时金融机构网上支付系统反馈有关支付信息，客户确认无误后进行支付确定，支付网关负责商户网上交易资金的清算，并根据商户提供的开户行、

账号等结账信息将网上消费款项汇总划入商户账户。

BCTY365 网上社区的在线支付是与中国工商银行合作来完成的。BCTY365 网上社区的在线支付操作步骤如下：

（1）登录网上社区，如图 22.41 所示。

（2）购买商品。在本页中，不但可以购买商品，还可以查看商品的详细信息和购物车中的商品信息，如图 22.42 所示。

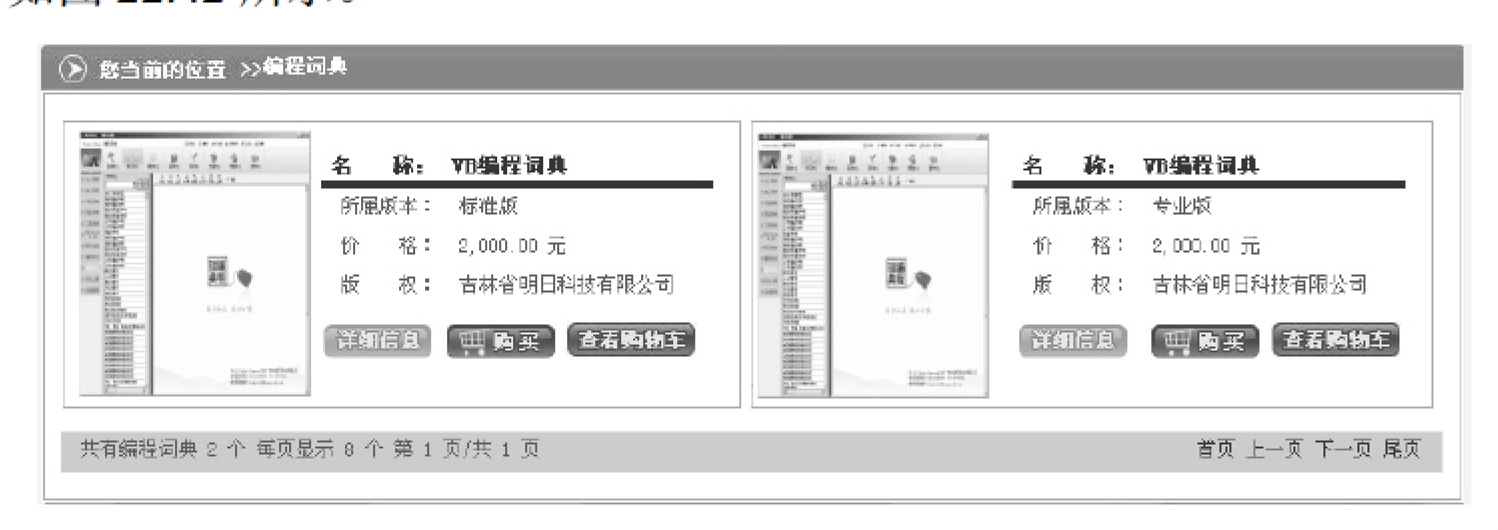


图 22.41 在线订购的操作页面



图 22.42 购买商品操作页面

（3）进入购物车操作页面，如图 22.43 所示。在该页面中，可以修改购物数量、删除指定商品、清空购物车、继续购物和统计购买商品的金额，也可以单击“结算”按钮进入到商品结算页面。

（4）进入到购物结算页面，填写收货人的详细信息，确认后提交该数据，如图 22.44 所示。

商品名称	单价（元）	数量（个）	操作
VB编程词典	2,000.00	1	更改数目 | 删除该项

<< 继续购买 清空购物车 >> 商品金额总计：2,000.00 元 去结算

图 22.43 购物车操作页面



图 22.44 填写收货人的详细信息

（5）订单确认。订单确认以后，就可以提交订单，准备进行网上支付，如图 22.45 所示。

（6）进行网上支付。在这里可以选择工行网上支付，也可以选择取消该订单，如图 22.46 所示。


吉林省明日科技有限公司-编程词典订单 2007-12-13 11:54:39

订单号：07121311451945
收货人：潘**　　邮编：130000
移动电话：1360433****　　固定电话：0431-84978981
联系地址：长春市

商品名称	单价（元）	数量	小计（元）
VB编程词典	2,000.00	1	2,000.00

商品总计：2,000.00 元
邮费：240.10 元
你需要支付的金额总计为：2,240.10 元

打 印　打印预览　提交订单

图 22.45 订单确认

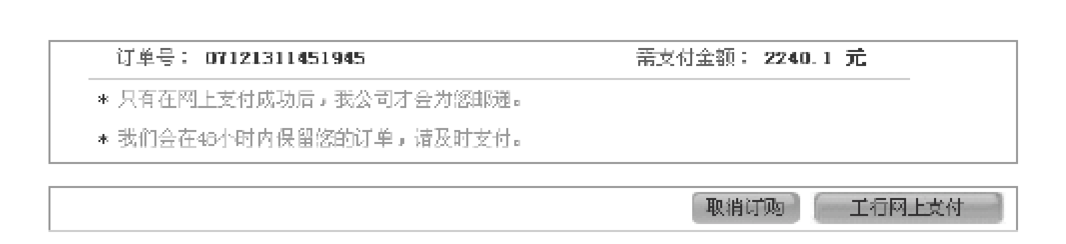
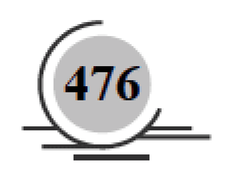


图 22.46 执行网上支付



接下来的操作在工行 B2C 支付页面上进行。首先网上社区按照工商银行 B2C 订单数据规范形成提交数据，并使用工商银行提供的 API 和商户证书对订单数据签名，形成 form 表单返回客户浏览器，表单 action 地址指向工商银行接收商户 B2C 订单信息的 servlet，然后在客户确认使用工行网上支付后，提交此表单到工商银行，最后工行网银系统接收此笔 B2C 订单，对订单信息和商户信息进行检查，通过检查则显示工行 B2C 支付页面。

客户通过工行 B2C 支付页面实现网上支付，商户查询网上银行的账户，如果货款已经到账，则根据客户指定的方式将货物送达客户手中。

上述内容就是网上社区系统的在线支付流程，涉及到工商银行的操作内容这里不做讲解。这里主要讲解一下如何将订单信息提交到工商银行。该项操作主要通过 shopping_tjdd.php 文件来实现，首先从数据库中读取订单信息，然后将订单信息输出，最后创建“取消订购”和“工行网上支付”两个超链接，通过 JavaScript 脚本来调用不同的执行文件。其关键代码如下：

```
<?php	include_once("conn/conn.php"); include_once("top.php"); ?>	//连接数据库和网站的头文件
<!-->省略了部分代码<!-->
<?php
❶	$ddnumber=base64_decode($_GET["ddno"]);	//对获取的订单编号进行 base64 解码
//获取该订单的金额信息
$sql=mysql_query("select * from tb_dd where ddnumber='".$ddnumber."'",$conn);
$info=mysql_fetch_array($sql);
$amount=$info["totalprice"];
❷	$amount=str_replace(",","",number_format($amount,2));	//修改数字的输出格式
$amount=str_replace(".","",number_format($amount,2));	//修改数字的输出格式
?>
…	//省略部分 HTML 代码
<table width="630" border="0" align="center" cellpadding="0" cellspacing="0">
	<tr><td width="159"> 
<?php
	$sql=mysql_query("select totalprice from tb_dd where ddnumber='".base64_decode($_GET["ddno"])."'",$conn);
	$info=mysql_fetch_array($sql);
		echo "<font color=red><strong>".$info["totalprice"]." 元</strong></font>";
?>		</td>
	</tr>
</table>
…	//省略部分 HTML 代码
<script language="javascript">
//打印订单
❸	function openprintwindow(x,y,z){
	window.open("printwindow.php?ddno="+x+"&pv="+z,"newframe","top=200,left=200,width=635,height="+(
230+20*y)+",menubar=no,location=no,toolbar=no,scrollbars=no,status=no");
	}
</script>
…	//省略部分 HTML 代码
<table width="630" height="25" border="0" align="center" cellpadding="0" cellspacing="0">
	<tr>
		<!-- -------------------------------------------取消该订单----------------------------------------- -->
		<td width="75"><img src="images/bg_14(14).jpg" width="69" height="20" style="cursor:hand"
onclick="javascript:if(window.confirm('如果取消该订单，则该订单将被删除，您需要重新购买！
')==true){window.location.href='deletedd.php?ddno=<?php echo $_GET["ddno"];?>';}"/></td>
		<!-- -------------------------------------------执行网上支付----------------------------------------- -->
		<td width="125"><img src="images/bg_14(15).jpg" width="119" height="20"
onclick="javascript:window.location.href='ddform.php?orderid=<?php echo
base64_decode($_GET["ddno"]);?>&amount=<?php echo $amount;?>&orderDate=<?php echo date("Ymdhis");?>';"
```

```
style="cursor:hand"/></td>
        </tr>
</table>
<!-- --------------------------------------------------省略了部分代码------------------------------------------- -->
<?php    include_once("bottom.php"); ?>                                    //包含网站的尾文件
```

说明

❶ base64_decode()：PHP 实现对 base64 编码的字符进行解码。PHP 实现字符串的 base64 编码通过 base64_encode()函数。

❷ str_replace()：实现字符串的替换。该函数的语法如下：

mixed str_replace (mixed search, mixed replace, mixed subject , int &count)

str_replace()将所有在参数 subject 中出现的 search 以参数 replace 替换，参数&count 表示替换字符串执行的次数。

❸ openprintwindow()：JavaScript 脚本中自定义的函数，用于执行订单的打印操作。

22.12 小　　结

本章从项目开发的实际角度出发，详细地讲解了 BCTY365 网上社区系统的开发过程，其中以系统的整体开发流程为主线，重点介绍技术支持、在线订购、社区论坛和编程词典等几个大模块的实现方法，并且对管理员权限设置、帖子置顶设置和在线支付技术进行讲解。



高级应用

- 第 23 章　Smarty 模板技术
- 第 24 章　ThinkPHP 框架
- 第 25 章　Zend Framework 框架
- 第 26 章　综合实例（五）——电子商务网站

第23章

Smarty 模板技术

（视频讲解：70 分钟）

目前网络上针对 PHP 的模板数不胜数。作为最早的 MVC 模板之一，Smarty 在功能和速度上处于绝对的领先地位。那么，Smarty 的特点是什么？它是如何完成代码分离的？

通过阅读本章内容，你可以：

- 了解什么是 MVC，什么是 Smarty
- 掌握 Smarty 的特点
- 掌握 Smarty 模板的安装和配置的方法
- 掌握 Smarty 模板设计的方法
- 掌握 Smarty 程序设计的方法
- 熟悉 Smarty 模板的应用

23.1　Smarty 简介

Smarty 是 PHP 中的一个模板引擎，是众多 PHP 模板中最优秀、最著名的模板之一。Smarty 模板引擎将 PHP 程序直接生成模板文件，最终浏览器中读取的就是 Smarty 模板文件，并且 Smarty 能够对模板文件进行判断，如果是第一次或者模板已经改变，则重新生成模板文件，否则将直接执行原模板文件。Smarty 模板引擎的运行流程如图 23.1 所示。

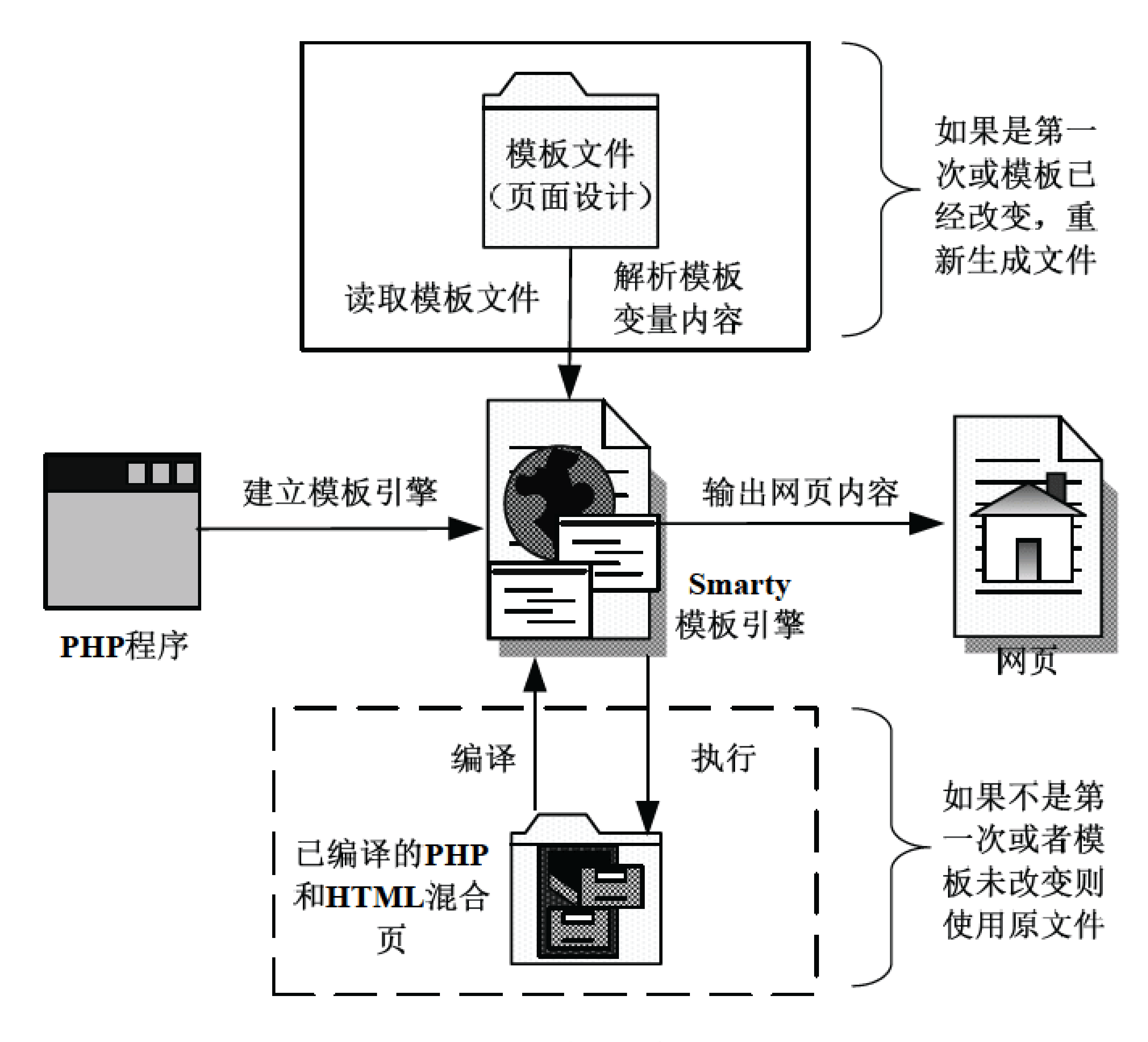


图 23.1　Smarty 模板引擎运行流程图

23.1.1　Smarty 模板引擎

当动态网页开始风行时，网站设计师们对 HTML 中嵌入脚本语言又提出了更高的要求，对这种方式很不满意。因为无论是微软的 ASP 还是开放源码的 PHP，都属于服务器端的 HTML 嵌入式脚本语言，作为一个网站设计师必须既懂程序设计，又懂页面设计。但在通常情况下，如果程序设计是高手，那么在页面设计上必然差一些，反之也是如此，正所谓“鱼和熊掌不可兼得”。由此有网站设计师设想，如果能将网页设计与程序开发分离，效果是否会更好呢？因此模板引擎应运而生。

开发模板引擎的目的是实现程序设计与网页设计的逻辑分离，使程序设计者可以专注于程序功能的开放；而网页设计师专注于页面的设计，让网页看起来更加具有专业性。

Smarty 是一个使用 PHP 编写的应用于 PHP 的模板引擎，它将一个应用程序分成两部分：视图和逻辑控制。简单地讲，就是将 UI（用户界面）和 PHP code（PHP 代码）分离。这样，程序员在修改程序时不会影响到页面设计，而美工在重新设计或修改页面时也不会影响程序逻辑。

23.1.2 Smarty 与 MVC

Smarty 的这种开发模式是基于 MVC 框架的概念。MVC（Model-View-Controller），即模型-视图-控制器。MVC 框架将一个应用程序定义成视图、控制器、模型 3 部分。

- ☑ 模型：对接收过来的信息进行处理，并将处理结果回传给视图。例如，如果用户输入信息正确，那么将给视图一个命令，允许用户进入主页面；反之，则拒绝用户的操作。
- ☑ 视图：就是提供给用户的界面。视图只提供信息的收集及显示，不涉及处理。如用户登录，登录界面就是视图，只提供用户登录的用户名和密码输入框（也可以有验证码、安全问题等信息），至于用户名和密码的对与错，这里不去处理，直接传给后面的控制部分。
- ☑ 控制器：负责处理视图和模型的对应关系，并将视图收集的信息传递给对应的模型。例如，当用户输入用户名和密码后提交。这时，控制部分接收用户的提交信息，并判断这是一个登录操作，随后将提交信息转发给登录模块部分，也就是模型。

23.1.3 Smarty 特点

- ☑ 采用 Smarty 模板编写的程序可以获得最快的速度。注意，这是相对于其他模板而言。
- ☑ 可以自行设置模板定界符，如{}、{{}}、<!--{}-->等。
- ☑ 仅对修改过的模板文件进行重新编译。
- ☑ 模板中可以使用 if/elseif/else/endif。
- ☑ 内建缓存支持。
- ☑ 可自定义插件。

23.2 Smarty 的安装配置

视频讲解：光盘\TM\Video\第 23 章\Smarty 的安装配置.exe

23.2.1 Smarty 下载和安装

PHP 没有内置 Smarty 模板类，需要单独下载和配置，并且 Smarty 要求服务器上的 PHP 版本最低为 4.0.6。用户可以通过 http://smarty.net/download.php 网站下载最新的 Smarty 压缩包。本章使用的版本是 Smarty-2.6.26。

将压缩包解压后，有一个 libs 目录，其中包含了 Smarty 类库的核心文件，包括 smarty.class.php、smarty_Compiler.class.php、config_File.class.php 和 debug.html 4 个文件，还有 internals 和 plug-ins 两个目录。将 libs 目录复制到服务器根目录下，并为其重新命名。一般该目录的名称为 smarty、class 等，这里将 libs 文件夹重新命名为 Smarty。至此，Smarty 模板安装完毕。

注意 凡是在后面的章节中提到 Smarty 类包、Smarty 目录等，都是这个重新命名后的 Smarty 目录，即原 libs 目录。

23.2.2　Smarty 配置

Smarty 模板引擎的配置步骤如下。

（1）确定 Smarty 目录的位置。因为 Smarty 类库是通用的，每一个项目都可能会使用到它，所以将 Smarty 放到根目录下。因为本章的所有程序都放在 TM/23/文件夹下，所以将/23/作为临时的根目录，Smarty 就放到这个目录下。

（2）新建 4 个目录 templates、templates_c、configs 和 cache。因为目录 templates 存放的是项目的模板，所以有人喜欢将 templates 放到 Smarty 目录外。这两种方法没什么区别，只要设置的路径正确即可。

（3）创建配置文件。如果要应用 Smarty 模板，就一定要包含 Smarty 类库和相关信息。将配置信息写到一个文件中，用的时候只要包含配置文件就可以。这里要注意一点，配置文件中要使用绝对路径，因为服务器不会知道文件在第几层目录中被调用。配置文件完成后，保存到根目录下。不要忘记本章中所指的根目录是/23/。配置文件 config.php 的代码如下：

```
<?php
/*  定义服务器的绝对路径  */
define('BASE_PATH','E:\AppServ\www\\');
/*  定义 Smarty 目录的绝对路径  */
define('SMARTY_PATH','TM\23\Smarty\\');
/*  加载 Smarty 类库文件  */
require BASE_PATH.SMARTY_PATH.'Smarty.class.php';
/*  实例化一个 Smarty 对象  */
$smarty = new Smarty;
/*  定义各个目录的路径  */
$smarty->template_dir = BASE_PATH.SMARTY_PATH.'templates/';
$smarty->compile_dir = BASE_PATH.SMARTY_PATH.'templates_c/';
$smarty->config_dir = BASE_PATH.SMARTY_PATH.'configs/';
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'cache/';
/*  定义定界符  */
//$smarty->left_delimiter = '<{';
//$smarty->right_delimiter = '}>';
?>
```

上述配置文件的参数说明如下。

- ☑ BASE_PATH：指定服务器的绝对路径。
- ☑ SMARTY_PATH：指定 smarty 目录的绝对路径。
- ☑ require()：加载 Smarty 类库文件 Smarty.class.php。
- ☑ $smarty：实例化 Smarty 对象。
- ☑ $smarty->template_dir：定义模板目录存储位置。
- ☑ $smarty-> compile_dir：定义编译目录存储位置。
- ☑ $smarty-> config_dir：定义配置文件存储位置。
- ☑ $smarty-> cache_dir：定义模板缓存目录。
- ☑ $smarty->left_delimiter：定义 Smarty 使用的开始定界符。
- ☑ $smarty->right_delimiter：定义 Smarty 使用的结束定界符。

注意　有关定界符的使用，开发者可以指定任意的格式，也可以不指定。使用 Smarty 默认的定界符“{”和“}”。

到此，Smarty 的配置讲解完毕。至于将配置文件存储在什么位置，可以根据实际情况而定。

23.2.3 第一个 Smarty 程序

视频讲解：光盘\TM\Video\第 22 章\第一个 Smarty 程序.exe

介绍 Smarty 的下载、安装和配置方法后，接下来就创建一个 Smarty 程序，通过具体的操作来了解 Smarty 的应用。

例 23.1 创建第一个 Smarty 应用实例，初步了解 Smarty 的使用过程。**（实例位置：光盘\TM\Instance\23\23.1）**

（1）在服务器根目录下的/TM/23/文件夹下，新建一个文件夹，命名为 23.1。

（2）复制 Smarty 到目录 23.1 下。在 Smarty 目录下新建 4 个目录，分别是 templates、templates_c、configs 和 cache。此时，例 23.1 的目录结构如图 23.2 所示。

（3）新建一个.html 静态页，输入数据。输入完毕后将文件保存到刚刚新建的 templates 目录下，并命名为 index.html。实例代码如下：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>{ $title }</title>
</head>
<body>
    {$content}
</body>
</html>
```

说明 代码中加粗的部分就是 Smarty 标签，大括号“{}”为标签的定界符，$title 和$content 为变量。

（4）回到上级目录，在目录 23.1 下新建一个.php 文件，使用 Smarty 变量和方法对文件进行操作，输入完毕后保存为 index.php。其代码如下：

```
<?php
/*  定义服务器的绝对路径  */
define('BASE_PATH',$_SERVER['DOCUMENT_ROOT']);
/*  定义 Smarty 目录的绝对路径  */
define('SMARTY_PATH','\TM\\23\23.1\Smarty\\');
/*  加载 Smarty 类库文件     */
require BASE_PATH.SMARTY_PATH.'Smarty.class.php';
/*  实例化一个 Smarty 对象  */
$smarty = new Smarty;
/*  定义各个目录的路径      */
$smarty->template_dir = BASE_PATH.SMARTY_PATH.'templates/';
$smarty->compile_dir = BASE_PATH.SMARTY_PATH.'templates_c/';
$smarty->config_dir = BASE_PATH.SMARTY_PATH.'configs/';
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'cache/';
/*  使用 Smarty 赋值方法将指定数据发送到模板中  */
$smarty->assign('title','第一个 Smarty 程序');
$smarty->assign('content','Hello,Welcome to study\'Smarty\'!');
/*  显示模板  */
$smarty->display('index.html');
?>
```

这是 Smarty 运行最关键的步骤，主要进行了两项设置和两步操作。

☑ 加载 Smarty 类库。也就是加载 Smarty.class.php 文件，这里使用的是绝对地址。为了稍后在配置其他路径时不用输入那么长的地址字串，之前还声明了两个常量：服务器地址常量和 Smarty 路径常量。两个常量连接起来就是 Smarty 类库所在的目录。

☑ 保存新建的 4 个目录的绝对路径到各自的变量。在第（2）步时，曾创建了 4 个目录。这 4 个目录各有各的用途，如果没有配置目录的地址，那么服务器默认的路径就是当前执行文件所在的路径。除了上面两项必须设置的变量外，还可以改变很多 Smarty 参数值，如开启/关闭缓存、改变 Smarty 的默认定界符等，这些变量将在 23.4.2 节中介绍。

☑ 给模板赋值。设置成功后，需要给指定的模板赋值。assign()就是赋值方法。

☑ 显示模板。一切操作结束后，调用 display()方法来显示页面。实际上，用户真正看到的页面是 templates 模板目录下的 index.html 模板文件；作为首页的 index.php，只是用来传递结果和显示模板；而在浏览器中运行的是 templates_c 文件夹下的文件。

打开 IE 浏览器，运行 index.php 文件。运行结果如图 23.3 所示。

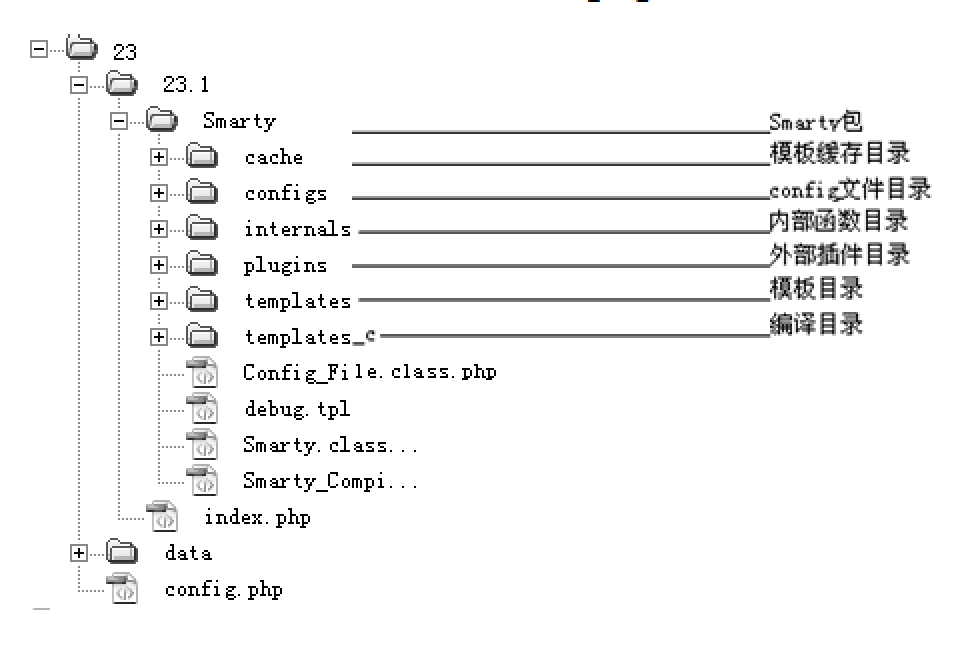


图 23.2　Smarty 包的目录结构

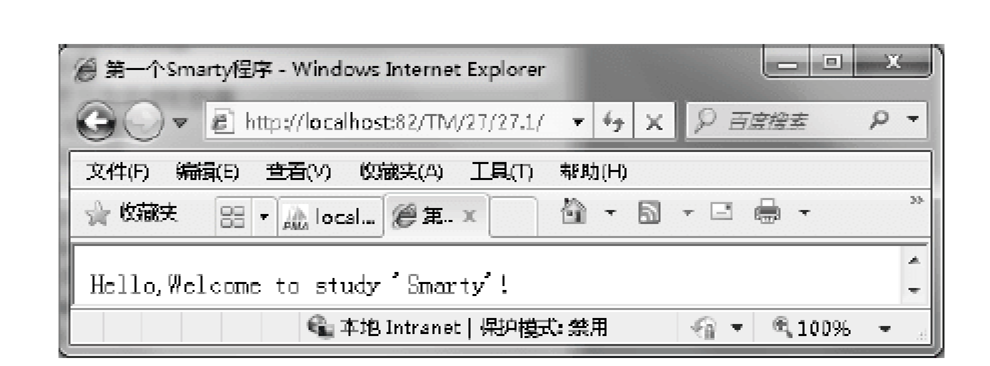


图 23.3　第一个 Smarty 程序

23.3　Smarty 模板设计

视频讲解：光盘\TM\Video\第 23 章\Smarty 模板设计.exe

Smarty 的特点是将用户界面和过程实现分离，让美工和程序员各司其职，互不干扰。这样，Smarty 类库也自然被分成两部分来使用，即 Smarty 模板设计和 Smarty 程序设计。两部分内容既相互独立，同时也有一部分共通。本节首先来学习 Smarty 模板设计。

23.3.1　Smarty 模板文件

Smarty 模板文件是由一个页面中所有的静态元素，加上定界符“{…}”组成的。模板文件统一存放的位置是 templates 目录下（模板文件的存储位置可以在配置文件中指定，可以根据个人习惯而定）。模板中不允许出现 PHP 代码段。Smarty 模板中的所有注释、变量、函数等都要包含在定界符内。

23.3.2　注释

Smarty 中的注释和 PHP 注释类似，都不会显示在源代码中。注释包含在两个星号“*”中间，格式如下：

```
{* 这是注释 *}
```

23.3.3 变量

Smarty 中的变量来自以下 3 个部分。

1．PHP 页面中的变量

使用方法和在 PHP 中是相同的，也需要使用“$”符号，略有不同的是对数组的读取。Smarty 中读取数组有两种方法，一种是通过索引获取，与 PHP 中相似，可以是一维，也可以是多维；另一种是通过键值获取数组元素，这种方法的格式与以前接触过的不同，使用符号“.”作为连接符，数组$arr = array{'object' => 'book','type' => 'comptuer','unit' => '本'}，如果想得到 type 的值，表达式的格式为$arr.type。这个格式同样适用于二维数组。

例 23.2 使用上述两种方法读取数组值。**（实例位置：光盘\TM\Instance\23\23.2）**

代码如下：

```
Smarty/templates/2/index.html 文件
<html>
<head>
{*  页面的标题变量$title  *}
<title>{ $title }</title>
</head>
<body>
购书信息：<p>
{*  使用索引取得数组的第一个元素值  *}
图书类别：{ $arr[0] }<br />
{*  使用键值取得第二个数组元素值  *}
图书名称：{ $arr.name }<br />
{*  使用键值取得二维数组的元素值  *}
图书单价：{ $arr.unit_price.price }/{ $arr.unit_price.unit }
</body>
</html>
index.php 文件
<?php
include_once'../config.php';
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'     $arr = array('computerbook','name' => 'PHP 开发实战宝典  ','unit_price' => array('price' => '￥65.00','unit' => '本'));
    $smarty->assign('title','使用 Smarty 读取数组');
    $smarty->assign('arr',$arr);
    $smarty->display('2/index.html');
?>
```

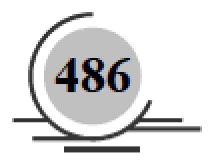
运行结果如图 23.4 所示。

2．保留变量

相当于 PHP 中的预定义变量。在 Smarty 模板中使用保留变量时，无需使用 assign()方法传值，直接调用变量名即可。Smarty 中常用的保留变量如表 23.1 所示。

表 23.1 Smarty 中常用的保留变量

保留变量名	说　明
get、post、server、session、cookie、request	等价于 PHP 中的$_GET、$_POST、$_SEVER、$_COOKIE、$_REQUEST
now	当前的时间戳等价于 PHP 中的 time()
const	用 const 包含修饰的为常量
config	配置文件内容变量。参见例 23.4



例 23.3　在模板文件中输出一些保留变量的值。（**实例位置：光盘\TM\Instance\23\23.3**）

代码如下：

```
templates/3/index.html 文件
{ *   设置标题名称   * }
<title>{ $title }</title>
<body>
{*   使用 get 变量获取 url 中的变量值(ex: http://localhost/tm/23/23.3/index.php?type=computer)   *}
变量 type 的值是：{ $smarty.get.type }<br />
当前路径为：{ $smarty.server.PHP_SELF}<br />
当前时间为：{$smarty.now}
</body>
index.php 文件
<?php
    include '../config.php';                              //载入配置文件
    $smarty->assign('title','Smarty 保留变量');          //向模板中赋值
    $smarty->display('3/index.html');                     //显示指定模板
?>
```

运行结果如图 23.5 所示。

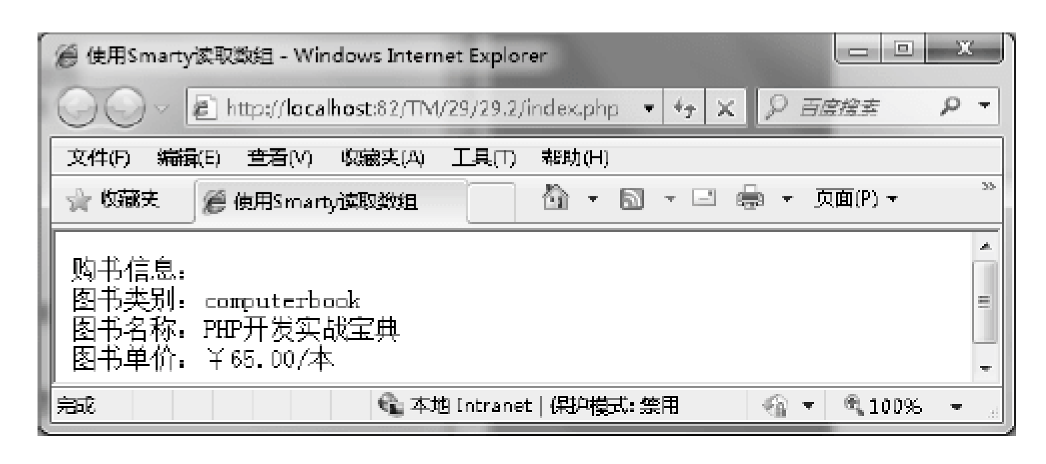


图 23.4　使用 Smarty 读取数组

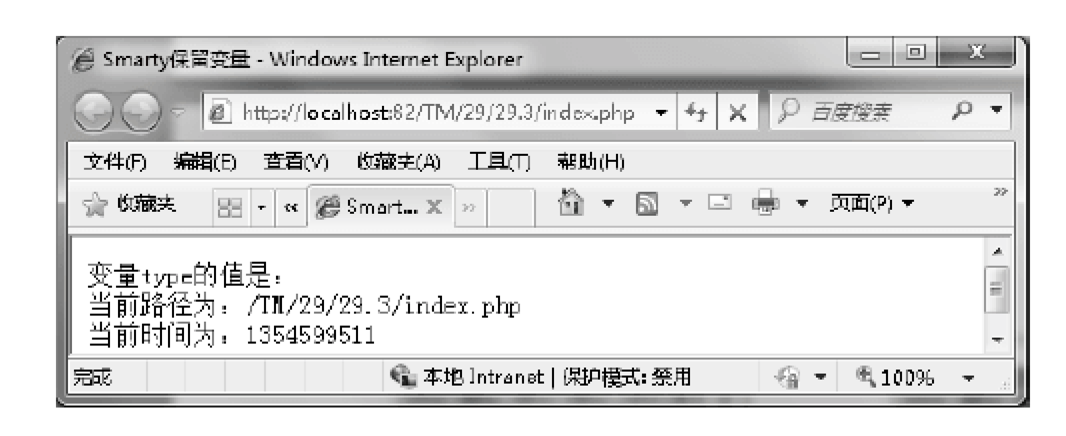


图 23.5　Smarty 保留变量

3．从配置文件中读取数据

Smarty 模板也可以通过配置文件来赋值。对于 PHP 开发人员来说，对配置文件的使用从安装服务器就开始了，对文件的格式也有了一个初步的了解。调用配置文件中变量的格式有以下两种：

☑　使用“#”号。将变量名置于两个“#”号中间，即可像普通变量一样调用配置文件内容。

☑　使用保留变量中的$smarty_config 来调用配置文件。

例 23.4　通过上面两种方法来调用配置文件 4.conf 的内容。（**实例位置：光盘\TM\Instance\23\23.4**）

代码如下：

```
Smarty/configs/4/4.conf 文件
title = "调用配置文件"
bgcolor = "#f0f0f0"
border = "5"
type = "计算机类"
name = "PHP 开发实战宝典"
templates/4/index.html 文件
{ config_load file="4/4.conf" }
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>{#title#}</title>
</head>
<body bgcolor="{#bgcolor#}">
<table border="{#border#}">
```

```
<tr>
        <td>{$smarty.config.type}</td>
        <td>{$smarty.config.name}</td>
</tr>
</table>
</body>
</html>
index.php 文件
<?php
        include_once '../config.php';
        $smarty->display('4/index.html');
?>
```

运行结果如图 23.6 所示。

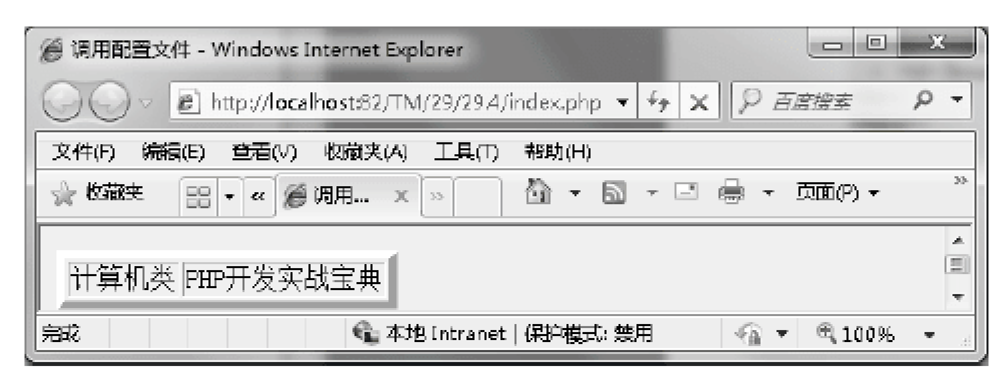


图 23.6　调用配置文件

23.3.4　修饰变量

23.3.3 节中介绍了如何在 Smarty 模板中调用变量。但有的时候，不仅要取得变量值，还要对变量进行处理。变量修饰的一般格式如下：

```
{variable_name|modifer_name: parameter1:…}
```

☑　variable_name 为变量名称。

☑　modifer_name 为修饰变量的方法名。变量和方法之间使用符号“|”分隔。

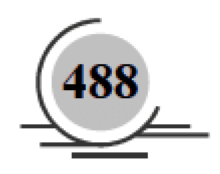
☑　parameter1 是参数值。如果有多个参数，则使用“:”分隔。

Smarty 提供了修饰变量的方法。常用的方法如表 23.2 所示。

表 23.2　修饰变量的常用方法和说明

方　法　名	说　　明
capitalize	首字母大写
count_characters:true/false	变量中的字符串个数。如果后面有参数 true，则空格也被计算，否则忽略空格
cat: "characters"	将 cat 中的字符串添加到指定字符串的后面
date_format: "%Y-%M-%D"	格式化日期和时间。等同于 PHP 中的 strftime()函数
default: "characters"	设置默认值。当变量为空时，将使用 default 后面的默认值
escape: "value"	用于字符串转码。value 值可以为 html、htmlall、url、quotes、hex、hexentity 和 javascript，默认为 html
lower	将变量字符串改为小写
nl2br	所有的换行符将被替换成 ，功能同 PHP 中的 nl2br()函数一样
regex_replace:"parameter1":"value2"	正则替换。用 value2 替换所有符合 parameter1 标准的字串
replace: "value1":"value2"	替换。使用 value2 替换掉所有 value1
string_format: "value"	使用 value 来格式化字符串。如果 value 为%d，则字符串被格式化为十进制数
strip_tags	去掉所有的 html 标签
upper	将变量改为大写

在对变量进行修饰时，不仅可以单独使用上面的方法，还可以同时使用多个。需要注意的是，在每个方法之间使用“|”分隔即可。



例 23.5　使用表 23.2 中的几个方法来修饰字符串。其他方法的使用，读者可以自行练习。**（实例位置：光盘\TM\Instance\23\23.5）**

实例代码如下：

```
Templates/5/index.html 文件
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>{$title}</title>
</head>
<body>
原文：{$str}
<p>
变量中的字符数（包括空格）：{$str|count_characters:true}
<br />
使用变量修饰方法后：{$str|nl2br|upper}
</body>
</html>
index.php 文件
<?php
	include_once "../config.php";
	$str1 = '这是一个实例。';
	$str2 = "\n 图书->计算机类->php\n 书名：《php 开发实战宝典》";
	$str3 = "\n 价格：￥86/本。";
	$smarty->assign('title','使用变量修饰方法');
	$smarty->assign('str',$str1.$str2.$str3.$str4);
	$smarty->display('5/index.html');
?>
```

运行结果如图 23.7 所示。

23.3.5　流程控制

Smarty 模板中的流程控制语句包括 if…elseif…else 条件控制语句和 foreach、section 循环控制语句。

1．if…elseif…else 语句

if 条件控制语句的使用和 PHP 中的 if 大同小异。需要注意的是，在 Smarty 模板中 if 必须以/if 为结束标记。if 语句的格式如下：

```
{if 条件语句 1}
	语句 1
{elseif 条件语句 2}
	语句 2
{else}
	语句 3
{/if}
```

上述条件语句中，除了可以使用 PHP 中的<、>、=、!=等常见运算符外，还可以使用 eq、ne、neq、gt、lt、lte、le、gte、ge、is even、is odd、is not even、is not odd、not、mod、div by、even by、odd by 等修饰词修饰。具体含义留给读者自己动手来理解。

例 23.6　使用条件判断语句选择不同的返回信息。**（实例位置：光盘\TM\Instance\23\23.6）**

代码如下：

```
templates/6/index.html 文件
```

```
<html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>{$title}</title>
<link rel='stylesheet' href="../css/style.css" />
</head>
<body>
<p>
{if $smarty.get.type == 'tm'}
欢迎光临，{$smarty.get.type}
{else}
对不起，您不是本站 VIP，无权访问此栏目。
{/if}
</body>
</html>
```

index.php 文件

```
<?php
	include_once "../config.php";
	$smarty->assign("title","if 条件判断语句");
	$smarty->display("6/index.html");
?>
```

运行结果如图 23.8 所示。

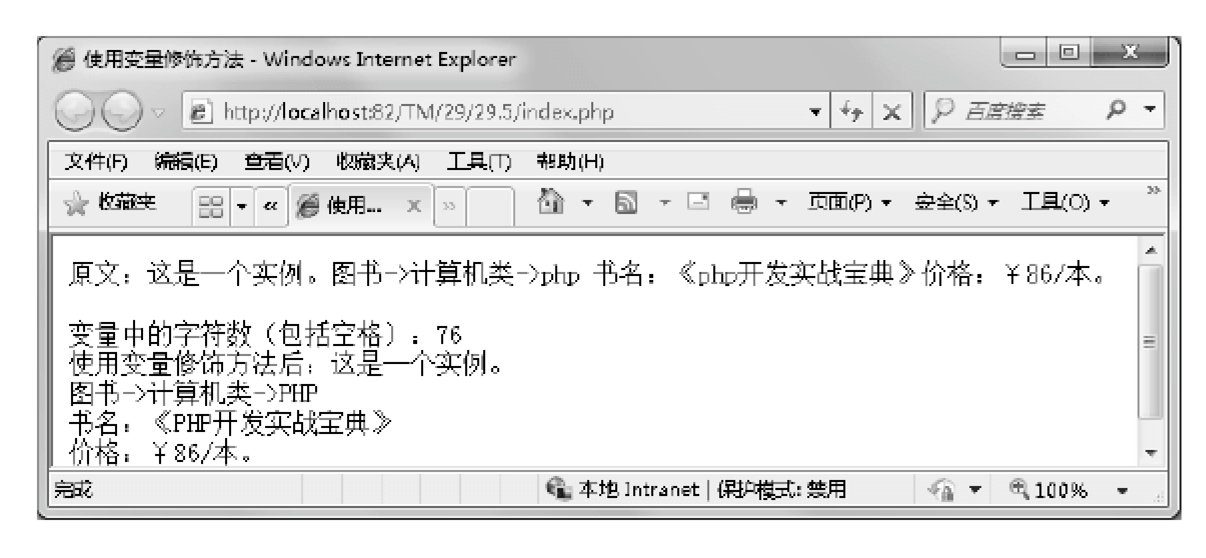


图 23.7　使用变量修饰方法

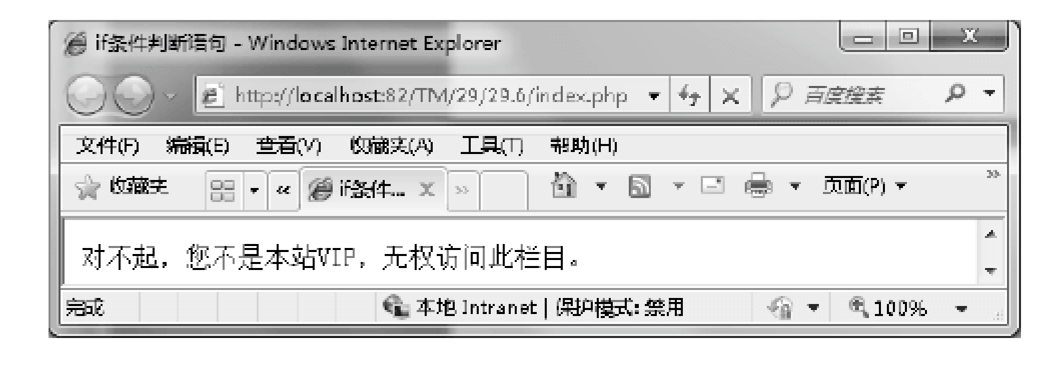


图 23.8　if 条件判断语句

2．foreach 循环控制

Smarty 模板中的 foreach 语句可以循环输出数组，与另一个循环控制语句 section 相比，在使用格式上要简单得多，一般用于简单数组的处理。foreach 语句的格式如下：

```
{foreach name=foreach_name key=key item=item from=arr_name}
…
{/foreach}
```

☑　name 为该循环的名称。
☑　key 为当前元素的键值。
☑　item 为当前元素的变量名。
☑　from 为该循环的数组。

其中，item 和 from 是必选参数，不可省略。

例 23.7　使用 foreach 语句，循环输出数组 infobook 的全部内容。**（实例位置：光盘\TM\Instance\23\23.7）**

代码如下：

templates/7/index.html 文件

```
<html>
```



```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>{$title}</title>
</head>
<body>
使用 foreach 语句循环输出数组。<p>
{foreach key=key item=item from=$infobook}
{$key} => {$item}<br />
{/foreach}
</body>
</html>
```

index.php 文件

```
<?php
	include_once '../config.php';
	$infobook = array('object'=>'book','type'=>'computer','name'=>'PHP 开发实战宝典','publishing'=>'清华大学出版社');
	$smarty->assign('title','使用 foreach 循环输出数组内容');
	$smarty->assign('infobook',$infobook);
	$smarty->display('7/index.html');
?>
```

运行结果如图 23.9 所示。

3．section 循环

Smarty 模板中的另一个循环语句是 section，该语句可用于比较复杂的数组。section 语句的语法如下：

```
{section name="sec_name" loop=$arr_name start=num step=num}
```

☑　name 为该循环的名称。
☑　loop 为循环的数组。
☑　start 表示循环的初始位置，如 start=2，则说明循环是从 loop 数组的第二个元素开始的。
☑　step 表示步长，如 step=2，则循环一次后数组的指针将向下移动两位，依此类推。

例 23.8　使用 section 语句循环输出一个二维数组。（**实例位置：光盘\TM\Instance\23\23.8**）

代码如下：

templates/8/index.html 文件

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>{$title}</title>
<link rel="stylesheet" href="../css/style.css" />
</head>
<body>
<table width="100" border="0" align="left" cellpadding="0" cellspacing="0">
{section name=sec1 loop=$obj}
	<tr>
		<td colspan="2">{$obj[sec1].bigclass}</td>
	</tr>
	{section name=sec2 loop=$obj[sec1].smallclass}
	<tr>
		<td width="25"> </td>
		<td width="75">{$obj[sec1].smallclass[sec2].s_type}</td>
	</tr>
	{/section}
{/section}
```

```
</table>
</body>
</html>
index.php 文件
<?php
	require "../config.php";
	$obj = array(array("id" => 1, "bigclass" => "计算机图书","smallclass" => array(array("s_id" => 1, "s_type" =>
"PHP"))),array("id" => 2, "bigclass" => "历史传记","smallclass" => array(array("s_id" => 2, "s_type" => "中国历史
"), array("s_id" => 3, "s_type" => "世界历史"))),array("id" => 3, "bigclass" => "电子小说","smallclass" => array
(array("s_id" => 4, "s_type" => "玄幻小说"),array("s_id" => 5, "s_type" => "言情小说"))));
	 $smarty->assign('title','section 循环控制');
	$smarty->assign("obj", $obj);
	$smarty->display("8/index.html");
?>
```

运行结果如图 23.10 所示。

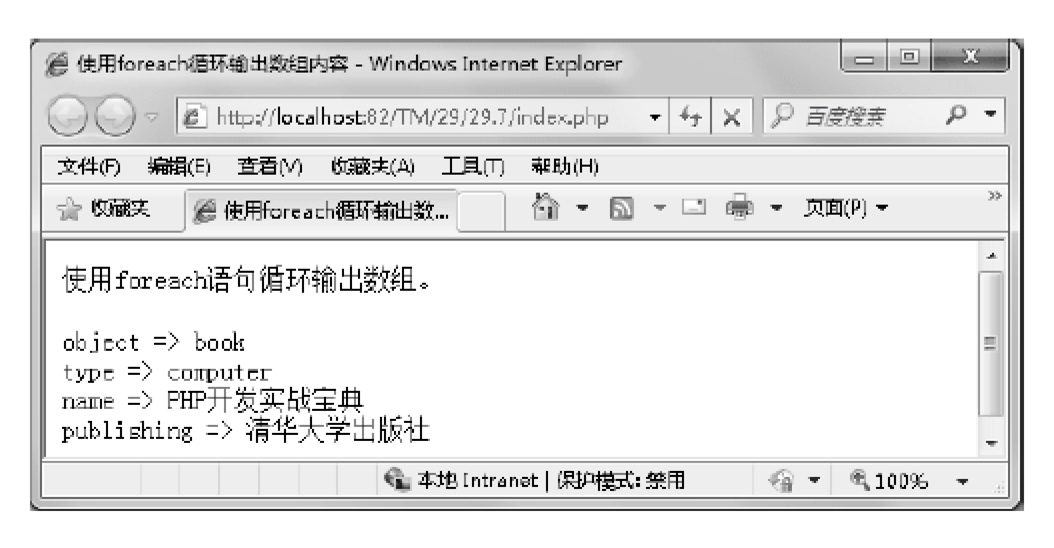


图 23.9　使用 foreach 语句循环输出数组内容

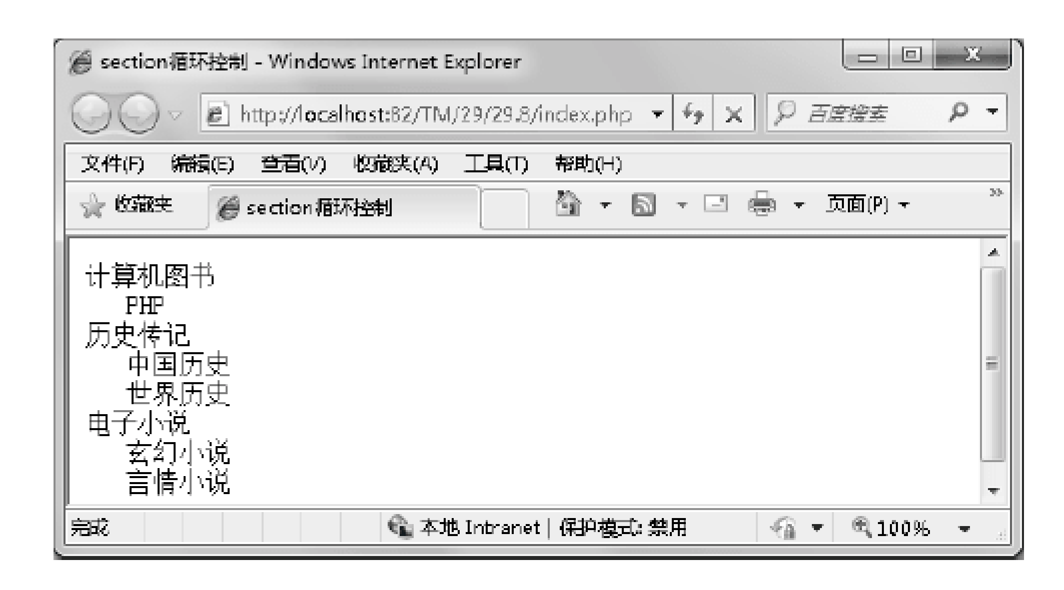


图 23.10　使用 section 循环控制输出数组

23.4　Smarty 程序设计

视频讲解：光盘\TM\Video\第 23 章\Smarty 程序设计.exe

通过前面的学习已经知道，在 Smarty 模板中，是不推荐有 PHP 代码段的，所有的 PHP 程序都要另写成文件。Smarty 程序的功能主要分两种：一种功能是和 Smarty 模板之间的交互，如 assign()、display()方法；另一种功能就是配置 Smarty 函数，如变量 template_dir、$config_dir 等。本节将学习 Smarty 程序设计的其他一些方法和配置参数。

23.4.1　Smarty 中的常用方法

1．assign

assign 用于在模板被执行时为模板变量赋值。语法如下：

```
{assign var=" " value=" "}
```

参数 var 是被赋值的变量名，参数 value 是赋给变量的值。

2．display

display 方法用于显示模板，需要指定一个合法的模板资源的类型和路径。还可以通过第二个可选参数指定一个缓存号，相关的信息可以查看缓存。

```
void display (string template [, string cache_id [, string compile_id]])
```



- ☑ template 指定一个合法的模板资源的类型和路径。
- ☑ cache_id 为可选参数，指定一个缓存号。
- ☑ compile_id 为可选参数，指定编译号。编译号可以将一个模板编译成不同版本使用，如针对不同的语言编译模板。编译号的另外一个作用是，如果存在多个$template_dir 模板目录，但只有一个$compile_dir 编译后存档目录，就可以为每一个$template_dir 模板目录指定一个编译号，以避免相同的模板文件在编译后互相覆盖。相对于在每一次调用 display()时都指定编译号，也可以通过设置$compile_id 编译号属性来一次性设定。

Smarty 中除了使用 assign 和模板交互外，还有一些比较常用的方法。方法名称和功能说明如表 23.3 所示。

表 23.3 Smarty 程序设计常用方法和说明

方 法 格 式	说 明
void append (string varname, mixed var [, boolean merge])	该方法向数组中追加元素
void clear_all_assign ()	清除所有模板中的赋值
void clear_assign (string var)	清除一个指定的赋值
void config_load (string file [, string section])	加载配置文件，如果有参数 section，说明只加载配置文件中相对应的一段数据
string fetch (string template)	返回模板的输出内容，但不直接显示出来
array get_config_vars ([string varname])	获取指定配置变量的值，如果没有参数，则返回一个所有配置变量的数组
array get_template_vars ([string varname])	获取指定模板变量的值，如果没有参数，则返回一个所有模板变量的数组
bool template_exists (string template)	检测指定的模板是否存在

23.4.2 Smarty 的配置变量

Smarty 中只有一个常量 SMARTY_DIR，用来保存 Smarty 类库的完整路径，其他的所有配置信息都保存到相应的变量中。这里将介绍包括前面章节中接触过的 template_dir 等变量的作用及设置。

- ☑ $template_dir：模板目录。模板目录用来存放 Smarty 模板，在前面的实例中，所有的.html 文件都是 Smarty 模板。模板的后缀没有要求，一般都定义为.tpl、.html 等。
- ☑ $compile_dir：编译目录。顾名思义，就是编译后的模板和 PHP 程序所生成的文件，默认路径为当前执行文件所在的目录下的 templates_c 目录。进入到编译目录，可以发现许多“%%…%%index.html.php”格式的文件。随便打开一个这样的文件可以发现，实际上 Smarty 将模板和 PHP 程序又重新组合成一个混编页面。
- ☑ $cache_dir：缓存目录。用来存放缓存文件。同样，在 cache 目录下可以看到生成的.html 文件。如果 caching 变量开启，那么 Smarty 将直接从这里读取文件。
- ☑ $config_dir：配置目录。该目录用来存放配置文件。例 23.4 中所用到的配置文件，就保存到这里。
- ☑ $debugging：调试变量。该变量可以打开调试控制台。只要在配置文件（config.php）中将$smarty->debugging 设为 true 即可使用。
- ☑ $caching：缓存变量。该变量可以开启缓存。只要当前模板文件和配置文件未被改动，Smarty 就直接从缓存目录中读取缓存文件而不重新编译模板。

23.5 实　　战

视频讲解：光盘\TM\Video\第 23 章\实战.exe

23.5.1 Smarty 模板中日期、时间的格式化输出

例 23.9 在 PHP 脚本中日期、时间的格式化输出最常用的就是 date()函数，那么在 Smarty 模板中该如何完成日期、时间的输出呢？这就是本例中要讲解的内容，通过 Smarty 模板中的 date_format 函数完成日期、时间的格式化输出。**（实例位置：光盘\TM\Instance\23\23.9）**

（1）创建 system 文件夹，封装 Smarty 模板的配置方法，其具体内容可以参考基础训练 1，这里不再赘述了。

（2）创建 index.php 文件，包含 Smarty 配置文件，指定 Smarty 的模板页 index.html。

（3）创建 Smarty 模板页 index.html，在 index.html 模板文件中插入 date_format 函数，输出系统的当前时间。其关键代码如下：

```
当前时间： {$smarty.now|date_format:" %A   %B - %e - %Y"}
<!--{$smarty.now|date_format}
{$smarty.now|date_format:"%A, %B %e, %Y":$times}
{$smarty.now|date_format:"%H:%M:%S"}
{$yesterday|date_format}
{$yesterday|date_format:"%A, %B %e, %Y"}
-->
```

运行结果如图 23.11 所示。

23.5.2 Smarty 模板中的页面设计

例 23.10 在 Smarty 模板页中同样可以将网页的头、尾和主文件进行分别存储，从而达到页面重用的效果。这就是本例要讲解的内容，在 Smarty 模板页中设计网页页面。**（实例位置：光盘\TM\Instance\23\23.10）**

代码如下：

（1）创建 system 文件夹，封装 Smarty 模板的配置方法，这里不再赘述。

（2）创建 index.php 文件，应用 switch 语句根据超链接传递的变量值，包含不同的 PHP 脚本文件，同时将对应的模板页名称赋给指定的模板变量。其代码如下：

```
<?php
require("system/system.inc.php");                          //包含 Smarty 模板的配置类
if(isset($_GET['caption'])){                                //判断超链接的变量值
      $caption=$_GET['caption'];
}else{
      $caption="";
}
switch ($caption){                                          //根据 switch 语句实现不同页面之间的调整
      case "reg";
            include "reg.php";                              //包含 PHP 文件
            $smarty->assign('admin_phtml','reg.html');      //将对应的静态页文件赋给指定的模板变量
      break;
      case "log";
```

```
        include "log.php";
        $smarty->assign('admin_phtml','log.html');
    break;
    case "from";
        include "for.php";
        $smarty->assign('admin_phtml','for.html');
    break;
    case "mes";
        include "mes.php";
        $smarty->assign('admin_phtml','mes.html');
    break;
    default:
        include "for.php";
        $smarty->assign('admin_phtml','for.html');
    break;
}
$smarty->assign("title","Smarty 模板中的页面设计--".$caption) ;        //创建模板变量
$smarty->display("index.html");                                         //指定模板页
?>
```

（3）创建 index.html 模板页，应用 include 函数载入页面的头文件（top.html）、尾文件（bottom.html）和主文件。其关键代码如下：

```
{$title}
{include file=top.html}
{include file=$admin_phtml}
{include file=right.html}
{include file=bottom.html}
```

（4）创建 include 函数加载的模板文件和动态 PHP 文件。

运行结果如图 23.12 所示。



图 23.11　Smarty 模板中日期的格式化输出

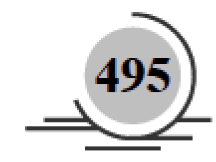


图 23.12　Smarty 模板中的页面设计

23.5.3　网站公告

例 23.11　应用 Smarty 模板技术，实现输出网站公告信息的功能。（**实例位置：光盘\TM\Instance\23\23.11**）

（1）创建 config.php 文件，配置 Smarty 模板。指定服务器的根目录、Smarty 模板在实例中的存储位置，以及各个文件夹的存储位置。其代码如下：



```
<?php
/*  定义服务器的绝对路径  */
define('BASE_PATH',$_SERVER['DOCUMENT_ROOT']);
/*  定义 Smarty 目录的绝对路径  */
define('SMARTY_PATH','\tm\23\23.11\Smarty\\');
/*  加载 Smarty 类库文件      */
require BASE_PATH.SMARTY_PATH.'Smarty.class.php';
/*  实例化一个 Smarty 对象  */
$smarty = new Smarty;
/*  定义各个目录的路径      */
$smarty->template_dir = BASE_PATH.SMARTY_PATH.'templates/';
$smarty->compile_dir = BASE_PATH.SMARTY_PATH.'templates_c/';
$smarty->config_dir = BASE_PATH.SMARTY_PATH.'configs/';
$smarty->cache_dir = BASE_PATH.SMARTY_PATH.'cache/';
?>
```

（2）创建 index.php 文件，从数据库中读取出公告信息，将公告信息存储到指定的模板变量中，并指定模板页。

```
<?php
    include_once "conn/conn.php";                          //连接数据库
    include_once "config.php";                             //调用配置文件
    $sql = "select id,title from tb_public order by id";   //编写查询语句
    $num = 4;
    $rst = $conn->SelectLimit($sql,$num);                  //执行查询操作
    $arr = $rst->GetAssoc();                               //获取结果集
    $smarty->assign('arr',$arr);                           //将返回的结果集存储到指定的 Smarty 模板变量中
    $smarty->display('index.html');                        //执行模板文件
?>
```

（3）在 smarty\templates 文件夹下创建 index.html 模板文件，输出模板变量中的公告标题信息。其关键代码如下：

```
  <td width="193" align="left" valign="top" class="exam">
         {foreach key=key item=item from=$arr}
          <a href="#" class="lk" onclick="return showme({$key},'showpub.php');" ><img src="images/man.JPG"
width="14" height="11" border="0" />{$item}</a><br />
         {/foreach}
     </td>
```

（4）创建 links.js 文件，定义 showme()函数，通过 JavaScript 脚本调用 showpub.php 文件，查看公告的详细内容。showme()函数的语法如下：

```
//JavaScript Document
function showme(key,wurl){
    var purl = wurl + "?id="+key;
    open(purl,'_blank','width=450 height=200',false);
    return false;
}
```

该函数的参数是指定公告信息的 ID 和调用的文件。

（5）创建 showpub.php 文件，根据传递的 ID 值，从数据库中查询出数据的详细信息，并将数据添加到指定的模板变量中，最后指定模板页。

```
<?php
    include_once 'conn/conn.php';
    include_once 'config.php';
    $id = $_GET['id'];
    $sql = "select * from tb_public where id = ".$_GET['id'];
    $rst = $conn->execute($sql);
```



```
	$arr = $rst->getAssoc();
	$smarty->assign('title','查看公告');
	$smarty->assign('arr',$arr[$id]);
	$smarty->display('showpub.html');
?>
```

（6）创建模板页 showpub.html，读取模板变量中的数据，输出公告的详细内容。其代码如下：

```
<table width="400" align="center" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td width="70%" height="25" align="center" valign="middle" class="left">标题：{$arr.title}</td>
    <td width="30%" height="25" align="center" valign="middle" class="right"> {$arr.addtime}</td>
  </tr>
  <tr>
    <td height="100" colspan="2" align="left" valign="top" class="all" style=" text-indent: 10px;"><br> 
{$arr.content}</td>
  </tr>
</table>
```

运行本实例，将直接输出网站公告的标题信息，单击某个具体的标题，将弹出一个新的窗口，输出具体公告的内容，如图 23.13 所示。

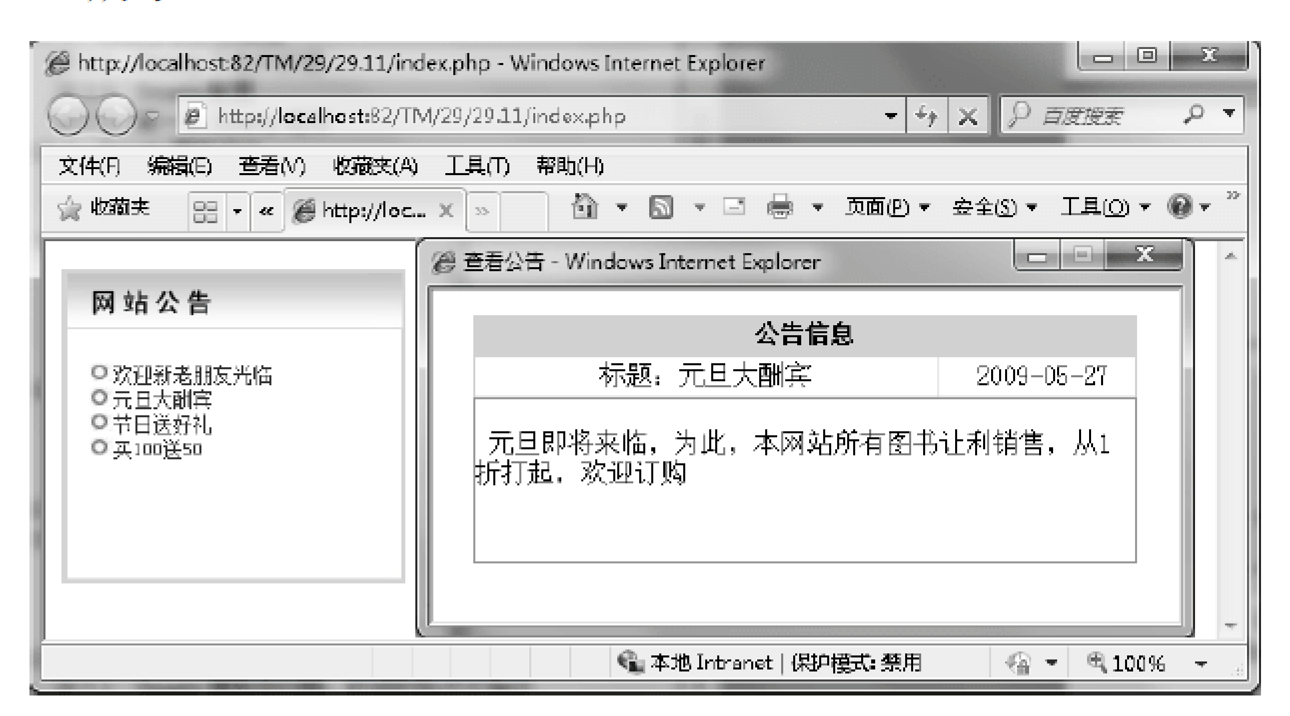


图 23.13　输出公告信息

23.5.4　Smarty 模板中应用正则表达式

例 23.12　通过 Smarty 模板中正则表达式的应用在 Smarty 模板页完成字符串的替换操作，将输出的会员名称“mr”替换为“明日科技”。**（实例位置：光盘\TM\Instance\23\23.12）**

regex_replace 变量 Smarty 模板中的正则表达式，依据正则表达式对指定的字符串进行匹配。其包括两个参数：第一个是指定的正则表达式；第二个是指定的替换文本。其应用示例如下：

```
{$name|regex_replace:"/[mr]/":"<font color='#FF0000'>明日科技</font>"}
```

在这个示例中，将模板变量$name 中的“mr”替换为“明日科技”。

（1）创建 system 文件夹，定义 Smarty 文件夹，存储编译目录、缓存目录；创建类文件 system.class.inc.php，定义数据库连接、操作类。创建 system.smarty.inc.php 文件，封装 Smarty 的配置类，定义类的实例化文件 system.inc.php，完成各个类的实例化操作，并返回操作对象。

（2）创建 index.php 文件。首先，通过 header()函数设置页面的编码格式，初始化 session 变量。然后，根据 session 变量判断当前用户是否具有访问权限。接着，如果具有访问权限，则应用 swicth 语句，根据超级链接传递的参数值，完成在不同页面之间的跳转操作，即通过 include 语句包含不同的动态 PHP 文件，同时将动态 PHP 文件对应的模板文件名称赋给模板变量。最后，指定模板页 main.html。

（3）创建模板页 main.html。首先，输出 PHP 动态页中定义的模板变量值，包括页面的标题（$title）、

当前时间（$dates）和当前页输出的模块类别（$type、$caption）。然后，通过 include 函数加载模板变量 $admin_phtml 传递的模板页。最后，为后台管理系统中每个功能模块创建热点链接，并通过超链接的参数传递数据，同时应用 Smarty 模板中的 escape 方法对传递的参数值进行编码。

（4）编辑 vip_look.php 文件，分页读取数据库中存储的会员数据，并且将分页读取返回的变量赋给模板变量，最终指定模板页。其关键代码如下：

```
<?php
require_once("system/system.inc.php");                                    //调用指定的文件
//执行查询语句，从数据库中读取商品信息
$info=$admindb->ExecSQL("select count(*) as total1 from tb_name ",$conn);
$total1=$info[0]['total1'];                                               //统计数据库中数据总数
if(empty($_GET['pages'])==true || is_numeric($_GET['pages'])==false){     //判断变量 pages 是否为空
      $page1=1;                                                           //如果变量为空，则赋值为 1
}else{                                                                    //如果不为空，则获取变量的值
      $page1=intval($_GET['pages']);
}
      $pagesize1=1;                                     //定义每页显示 3 条记录
      if($total1<$pagesize1){                           //判断如果数据库中数据小于每页显示的记录数
            $pagecount1=1;                              //则定义 pagecount 变量的值为 1
      }else{
            if($total1%$pagesize1==0){
                  $pagecount1=intval($total1/$pagesize1);  //用总的记录数除以每页显示的记录数，获取共有几页
            }else{
                  $pagecount1=intval($total1/$pagesize1)+1;
            }
      }
//将要输出的数据赋给 assign 模板变量
$smarty->assign("total1",$total1);
$smarty->assign("pagesize1",$pagesize1);
$smarty->assign("page1",$page1);
$smarty->assign("pagecount1",$pagecount1);
$array=$admindb->ExecSQL("select * from tb_name order by id desc limit ".($page1-1)*$pagesize1.
",$pagesize1" ,$conn );
if(!$array){
      $smarty->assign("iscommo","F");              //判断如果执行失败，则输出模板变量 iscommo 的值为 F
}else{
      $smarty->assign("iscommo","T");              //判断如果执行成功，则输出模板变量 iscommo 的值为 T
     $smarty->assign("arr",$array);                //定义模板变量 arraybbstell，输出数据库中数据
}
$smarty->assign(title,"会员浏览");
?>
```

（5）编辑 vip_look.html 模板页，通过 section 语句完成会员信息的分页输出，并且应用 regex_replace 变量将会员名称中的“mr”替换为“明日科技”。其关键代码如下：

```
{section name=id loop=$arr}
  <tr>
    <td align="center" bgcolor="#FFFFFF">{$arr[id].id}</td>
    <td align="left" bgcolor="#FFFFFF">{$arr[id].name|regex_replace:"/[mr]/":"<font color='#FF0000'>明日科技
</font>"}</td>
    <td align="left" bgcolor="#FFFFFF">{$arr[id].email}</td>
    <td align="left" bgcolor="#FFFFFF">{$arr[id].dates}</td>
  </tr>
  {/section}
```

运行结果如图 23.14 所示。

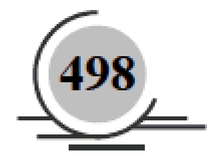




图 23.14　Smarty 模板中正则表达式的运用

23.5.5　if 语句判断当前用户权限

例 23.13　在动态 PHP 文件中，可以通过 session 变量的值判断当前用户是否具有访问权限，而在 Smarty 模板页中不可以使用 session 变量，那么该如何判断用户是否具有访问权限呢？这就是在本例中将要讲解的内容，应用 if 语句在 Smarty 模板页中判断用户是否具有访问权限。**（实例位置：光盘\TM\Instance\23\23.13）**

（1）创建 system 文件夹，封装 Smarty 模板的配置方法，创建存储编译文件、缓存文件和配置文件的文件夹。其具体内容可以参考基础训练 1，这里不再赘述了。

（2）创建 index.php 文件，包含 Smarty 配置文件，指定 Smarty 的模板页 index.html。

（3）创建 Smarty 模板页 index.html，设计用户登录页面，添加用户登录的表单元素，并且将用户的登录信息提交到 index_ok.php 文件。

（4）创建 index_ok.php 文件，获取表单中提交的数据，判断提交的用户名和密码是否正确，如果正确则通过 display()方法指定模板页 main.html，并且为模板变量 competence 赋值为 T；否则指定到模板页 main.html，为模板变量 competence 赋值为 F。其关键代码如下：

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                    //设置文件编码格式
include("system/system.inc.php");                                       //包含配置文件
if($_POST['Submit']!="" && $_POST['user']!="" && $_POST['pass']!=""){
        if($_POST['user']=="mr" && $_POST['pass']=="mrsoft"){
            $smarty->assign("competence","T");                          //为模板变量赋值
            $smarty->display('main.html');                              //指定模板页
        }else{
            $smarty->assign("competence","F");
            $smarty->display('main.html');
        }
}else{
```

```
        echo "<script>alert('用户名和密码不能为空！'); window.location.href='index.php';</script>";
}
?>
```

（5）创建 main.html 模板页。首先应用 if 语句对模板变量的值进行判断，如果值为 T 则输出本页内容；如果值为 F 则给出提示信息。其关键代码如下：

```
{if $competence=="T"}
//省略了部分内容
{/if}
{if $competence=="F"}
<script>alert('您没有权限访问，请重新登录');window.location.href="index.php";</script>
{/if}
```

运行结果如图 23.15 所示。

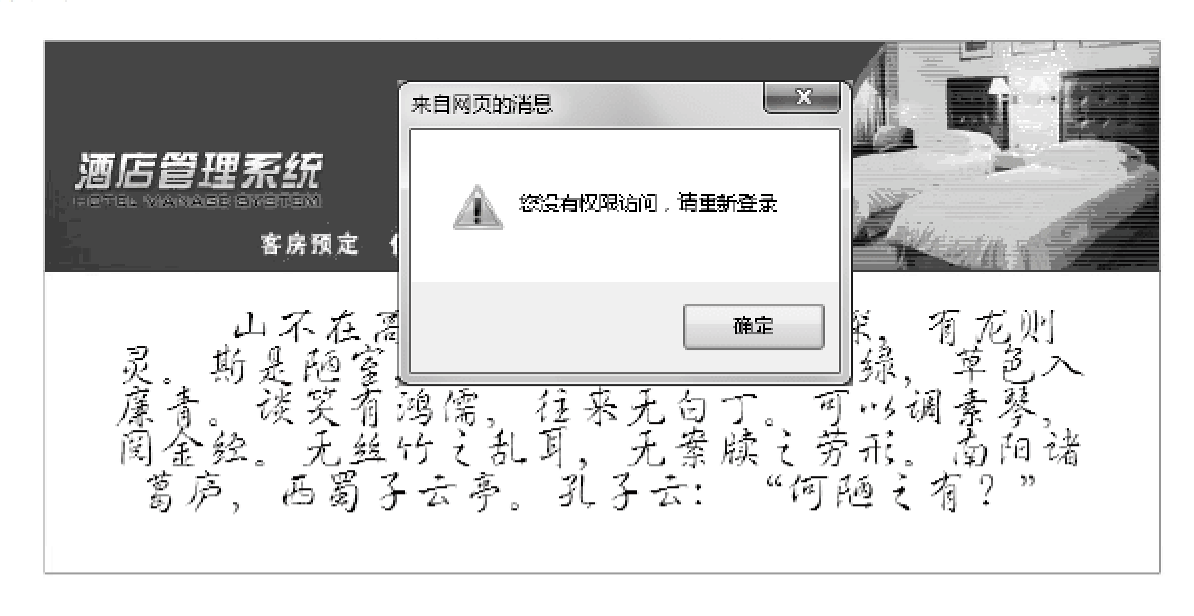


图 23.15　if 语句判断用户是否具备访问权限

23.6　小　　结

本章主要介绍了 Smarty 模板的安装、配置及使用，并且对 Smarty 的模板设计和程序设计进行了讲解，它们是两个相辅相成的内容，是应用 Smarty 模板的基础。在实战中讲解了 Smarty 的实际应用。希望通过本章的学习，读者能够掌握 Smarty 模板技术，并且能够将其灵活地运用到实际的网站开发中。

23.7　学习成果检验

1. 自定配置一个 Smarty 环境，并将 Smarty 定界符换成其他符号。（**实例位置：光盘\TM\Instance\23\23.14**）
2. 用 register_object 方法注册模板函数。（**实例位置：光盘\TM\Instance\23\23.15**）



第24章

ThinkPHP 框架

（视频讲解：153 分钟）

ThinkPHP 是一个免费开源、快速、简单的面向对象的轻量级 PHP 开发框架，遵循 Apache2 开源协议发布，是为了敏捷 Web 应用开发和简化企业级应用开发而产生的。

ThinkPHP 借鉴国外很多优秀的框架和模式，使用面向对象的开发结构和 MVC 模式，采用单一入口模式等，融合了 Struts 的 Action 思想和 JSP 的 TagLib（标签库）、RoR 的 ORM 映射和 ActiveRecord 模式，封装了 CURD 和一些常用操作，在项目配置、类库导入、模板引擎、查询语言、自动验证、视图模型、项目编译、缓存机制、SEO 支持、分布式数据库、多数据库连接和切换、认证机制和扩展性方面均有独特的表现。通过本章的学习，读者将对 ThinkPHP 框架有深入的认识，并且能够达到简单应用的程度。

通过阅读本章内容，你可以：

- 了解 ThinkPHP 概述
- 了解 ThinkPHP 的项目目录结构
- 了解 ThinkPHP 的控制器
- 了解 ThinkPHP 的视图
- 掌握 ThinkPHP 项目构建流程
- 掌握 ThinkPHP 的配置

24.1　ThinkPHP 简介

ThinkPHP 可以更方便和快捷地开发和部署应用。其不仅仅是企业级应用，任何 PHP 应用开发都可以从 ThinkPHP 的简单和快速的特性中受益。ThinkPHP 本身具有很多的原创特性，并且倡导大道至简，开发由我的开发理念，用最少的代码完成更多的功能，宗旨就是让 Web 应用开发更简单、更快速。

ThinkPHP 遵循 Apache2 开源许可协议发布，意味着可以免费使用 ThinkPHP，甚至允许把基于 ThinkPHP 开发的应用开源或商业产品发布/销售。

24.1.1　ThinkPHP 框架的特点

ThinkPHP 是一个性能卓越并且功能丰富的轻量级 PHP 开发框架。其宗旨就是让 Web 应用开发更简单、更快速。ThinkPHP 值得推荐的特性如下所示。

- ☑ 类库导入：ThinkPHP 是首先采用基于类库包和命名空间的方式导入类库，让类库导入看起来更加简单清晰，而且还支持冲突检测和别名导入。为了方便项目的跨平台移植，系统还可以严格检查加载文件的大小写。
- ☑ URL 模式：系统支持普通模式、PATHINFO 模式、REWRITE 模式和兼容模式的 URL 方式，支持不同的服务器和运行模式的部署，配合 URL 路由功能，可以随心所欲地构建需要的 URL 地址和进行 SEO 优化工作。
- ☑ 编译机制：独创的核心编译和项目的动态编译机制，有效减少了 OOP 开发中文件加载的性能开销。
- ☑ 查询语言：内建丰富的查询机制，包括组合查询、复合查询、区间查询、统计查询、定位查询、动态查询和原生查询，让数据查询简洁高效。
- ☑ 视图模型：轻松动态地创建数据库视图，多表查询不再烦恼。
- ☑ 分组模块：不用担心大项目的分工协调和部署问题，分组模块解决跨项目的难题。
- ☑ 模板引擎：系统内建了一款卓越的基于 XML 的编译型模板引擎，支持两种类型的模板标签，融合了 Smarty 和 JSP 标签库的思想，支持标签库扩展。通过驱动还可以支持 Smarty、EaseTemplate、TemplateLite、Smart 等第三方模板引擎。
- ☑ AJAX 支持：内置 AJAX 数据返回方法，支持 JSON、XML 和 EVAL 格式返回客户端，并且系统不绑定任何 AJAX 类库，可随意使用自己熟悉的 AJAX 类库进行操作。
- ☑ 缓存机制：系统支持包括文件方式、APC、Db、Memcache、Shmop、Eaccelerator 和 Xcache 在内的多种动态数据缓存类型，以及可定制的静态缓存规则，并提供了快捷方法进行存取操作。

24.1.2　环境要求

ThinkPHP 可以支持 Windows/UNIX 服务器环境，可运行于包括 Apache、IIS 在内的多种 Web 服务器，需要 PHP 5.0 及以上版本支持，支持 MySQL、MsSQL、PgSQL、Sqlite、Oracle 等数据库。

24.1.3　下载 ThinkPHP 框架

ThinkPHP 是一个免费开源、快捷、简单的 OOP 轻量级 PHP 开发框架。它遵循 Apache 2 开源协议发布，



是为了敏捷的企业级开发而产生的。获取 ThinkPHP 的方式有很多。

官方的网站为：http://thinkphp.cn。

SVN 的下载地址为：

完整版本 http://thinkphp.googlecode.com/svn/trunk；

核心版本 http://thinkphp.googlecode.com/svn/trunk/ThinkPHP。

1．什么是 MVC

MVC 是一种经典的程序设计理念，此模式将应用程序分为 3 个部分：模型层（Model）、视图层（View）、控制层（Controller），MVC 是这 3 个部分英文字母的缩写。

注意 MVC 设计模式产生的原因：应用程序中用来完成任务的代码——模型层（也叫“业务逻辑”），通常是程序中相对稳定的部分，重用率高；而与用户交互界面——视图层，却经常改变。如果因需求变动而不得不对业务逻辑代码修改，或者要在不同的模块中应用到相同的功能而重复的编写业务逻辑代码，不仅降低整体程序开发的进度，也会使未来的维护变得非常困难。因此将业务逻辑代码与外观分离，将会更方便地根据需求改进程序，这就是 MVC 设计模式。

在 PHP Web 开发中，MVC 设计模式的各自功能及相互关系如图 24.1 所示。

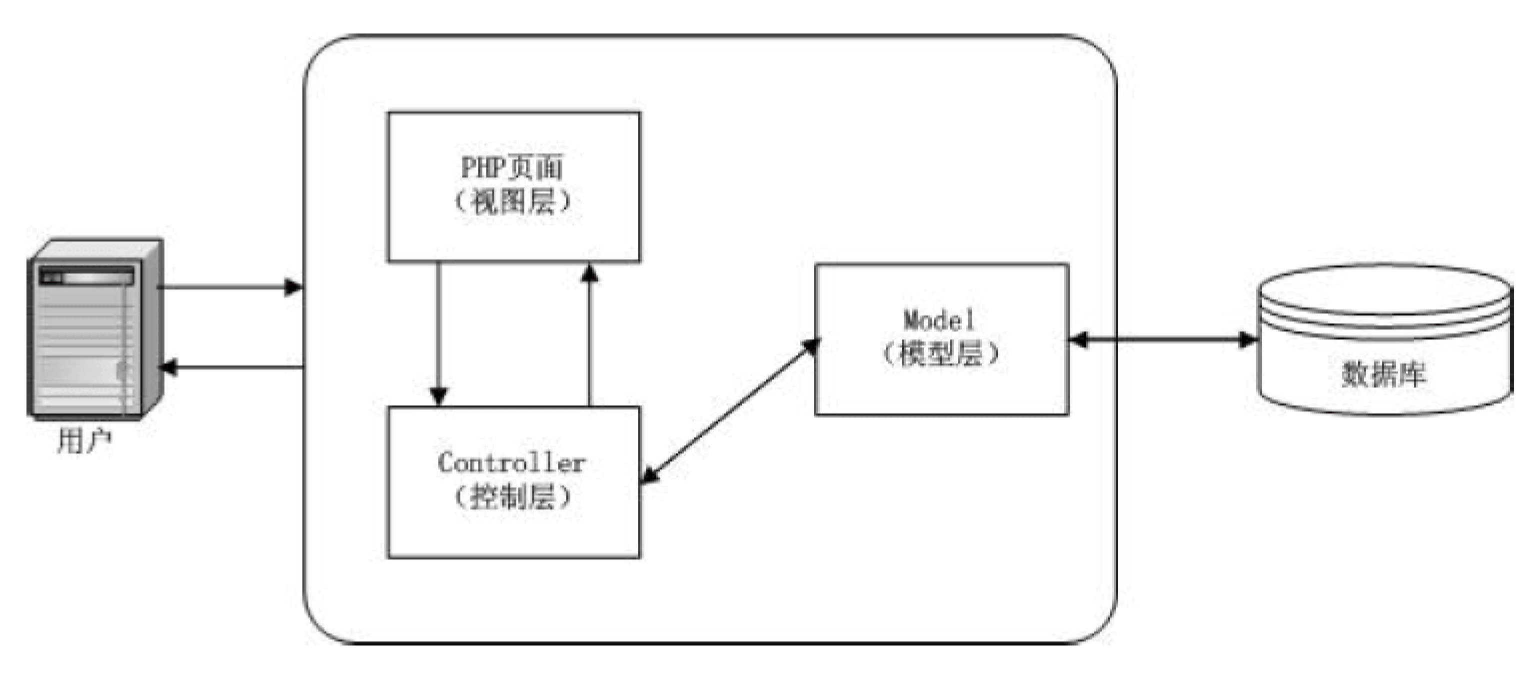


图 24.1　MVC 关系图

☑　模型层（Model）

模型层是应用程序的核心部分，它可以是一个实体对象或一种业务逻辑，它之所以称为模型，是因为它在应用程序中有更好的重用性和扩展性。

☑　视图层（View）

视图层提供应用程序与用户之间的交互界面，在 MVC 理论中，这一层并不包含任何的业务逻辑，仅提供一种与用户交互的视图。

☑　控制层（Controller）

控制层用于对程序中的请求进行控制，其作用就像国家的宏观调控，它可以选择调用哪些视图或者调用哪些模型。

2．什么是 CURD

CURD 是数据库操作的缩写词。也是几种数据库操作技术的缩写，“C”代表创建（Create）、“U”代表更新（Update）“R”代表读取（Read），“D”代表删除（Delete）操作。CURD 定义了用于处理数据的基本操作。之所以将 CURD 提升到一个技术难题的高度是因为完成一个涉及在多个数据库系统中进行 CURD 操作的汇总相关的活动，其性能可能会随数据关系的变化而有非常大的差异。

CURD 在具体的应用中并非一定使用 create、update、read 和 delete 字样的方法，但是它们完成的功能是一致的。例如，ThinkPHP 就是使用 add、save、select 和 delete 方法表示模型的 CURD 操作。

3. 什么是单一入口

单一入口通常是指一个项目或者应用具有一个统一（但并不一定是唯一）的入口文件，也就是说项目的所有功能操作都是通过这个入口文件进行的，并且往往入口文件是第一步被执行的。

单一入口的好处是项目整体比较规范，因为同一个入口，往往其不同操作之间具有相同的规则。另外一个方面就是单一入口控制较为灵活，因为拦截方便，如一些权限控制、用户登录方面的判断和操作可以统一处理。

24.2 ThinkPHP 架构

视频讲解：光盘\TM\Video\第 24 章\ThinkPHP 架构.exe

ThinkPHP 遵循简洁实用的设计原则，兼顾开发速度和执行速度的同时，也注重易用性。本节内容将对 ThinkPHP 框架的整体思想和架构体系进行详细说明。本章使用的版本是 ThinkPHP2.0。

24.2.1 ThinkPHP 的目录结构

ThinkPHP 框架中目录分为两部分：系统目录和项目目录。系统目录是下载的 ThinkPHP 框架类库本身的，如表 24.1 所示。

表 24.1 系统目录

目 录 名 称	主 要 作 用
Common	包含框架的一些公共文件、系统定义和惯例配置等
Lang	目录语言文件夹，目前 ThinkPHP 支持的语言包有简体中文、繁体中文、英文
Lib	系统的基类库目录
Tpl	系统的模板目录
Mode	框架模式扩展目录
Vendor	第三方类库目录

项目目录是用户实际应用的目录，如表 24.2 所示（ThinkPHP 采用自动创建文件夹的机制，当用户布置好 ThinkPHP 的核心类库后，编写运行入口文件，则相关应用到的项目目录就会自动生成）。

表 24.2 项目目录

目 录 名 称	主 要 作 用
index.php	项目入口文件
Common	项目公共目录，放置项目公共函数
Lang	项目语言包目录（可选）
Conf	项目配置目录，放置配置文件
Lib	项目基目录，通常包括 Action 和 Model 目录
Tpl	项目模板目录
Runtime	项目运行时目录，包括 Cache、Temp、Data 和 Log

24.2.2　自动生成目录

下面通过一个实例，讲解在 ThinkPHP 框架中如何自动生成项目目录。

例 24.1　创建名称为 TM 的项目，自动生成项目目录。（**实例位置：光盘\TM\Instance\24\24.1**）

其操作步骤如下。

（1）在网站根目录下创建文件夹，并命名为 TM。

（2）将 ThinkPHP 核心类库存储于 TM 目录下。

（3）编写入口文件 index.php，将其存储于 TM 目录下。index.php 文件代码如下：

```
<?php
define('THINK_PATH', 'ThinkPHP');                //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'TM');                        //定义项目名称
define('APP_PATH', '.');                         //定义项目路径
require(THINK_PATH."/ThinkPHP.php");             //加载框架入口文件
App::run();                                      //实例化一个网站应用实例
?>
```

在运行此文件前，查看 TM 项目的文件夹架构，如图 24.2 所示。

在 IE 浏览器中运行此项目，将输出如图 24.3 所示的运行结果，此为 ThinkPHP 提供的测试内容。此时再次查看例 24.1 的项目文件夹，如图 24.4 所示，在项目根目录下自动生成项目目录。



图 24.2　项目文件夹架构

图 24.3　已连接到 ThinkPHP 框架

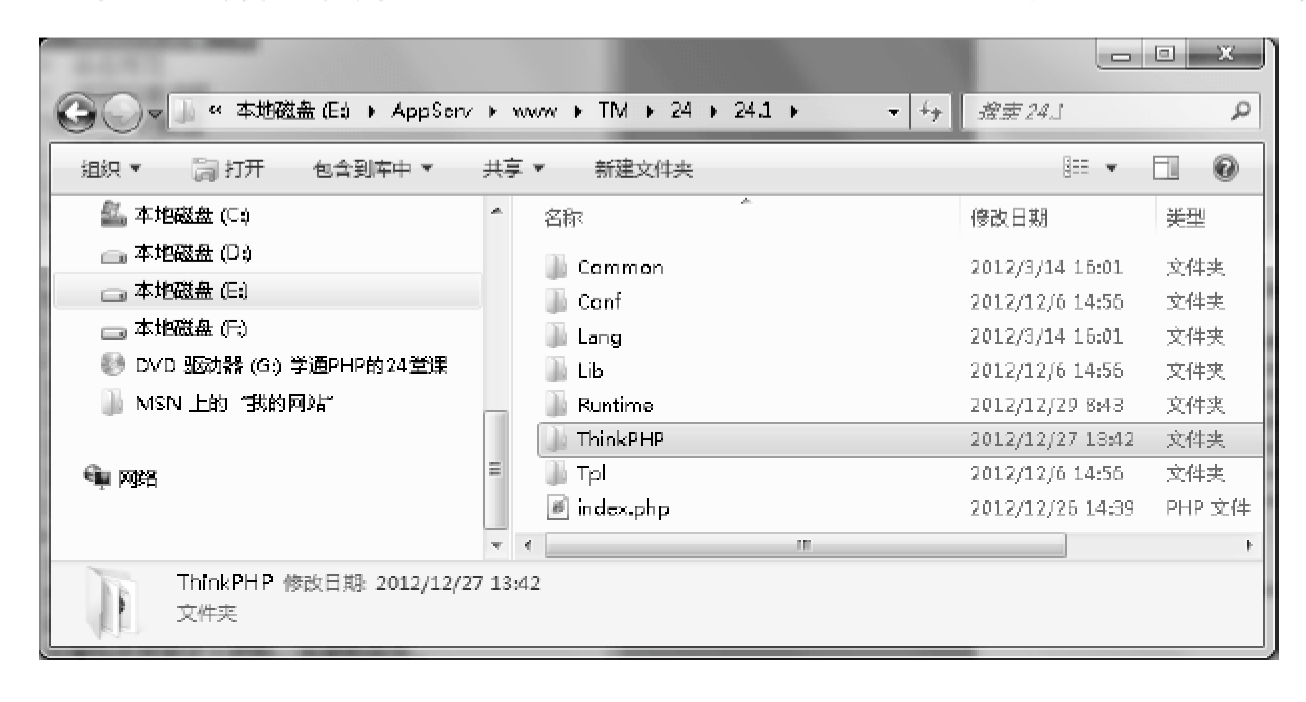


图 24.4　自动生成的项目目录

24.2.3　项目目录部署方案

在实际开发过程中，目录结构往往由于项目的复杂而变得复杂。这里向大家推荐两套标准的目录部署方案。

方案一如图 24.5 所示。

方案二采用分组模块，如图 24.6 所示。

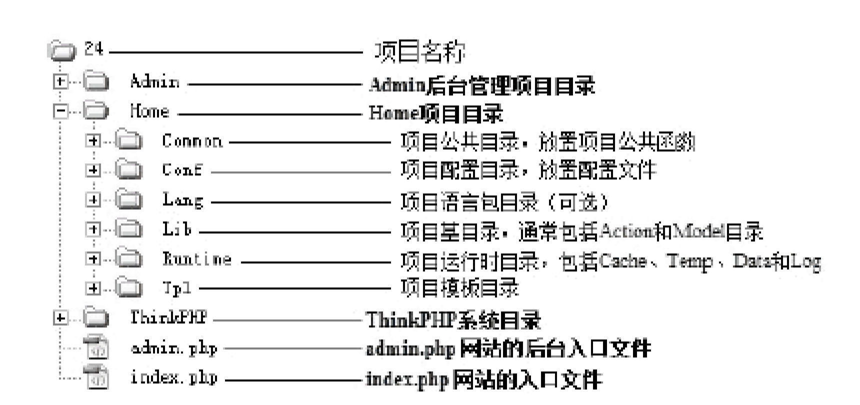


图 24.5　项目部署方案一



图 24.6　项目部署方案二

这样部署的好处是系统目录和项目目录可以存储于非 Web 访问目录下面，网站目录下面只需放置 Public 公共目录和 index.php 入口文件（如果是多个项目的话，每个项目的入口文件都需要放到 Web 目录下面），从而提高网站的安全性。

24.2.4　命名规范

ThinkPHP 框架有其自身的一定规范，要应用 ThinkPHP 框架开发项目，那么就要尽量遵守它的规范。下面就介绍一下 ThinkPHP 的命名规范。

☑ 类文件都是以.class.php 为后缀（这里指的是 ThinkPHP 内部使用的类库文件，不代表外部加载的类库文件），使用驼峰法命名，并且首字母大写，如 DbMysql.class.php。
☑ 函数、配置文件等其他类库文件之外的一般是以.php 为后缀（第三方引入的不做要求）。
☑ 确保文件的命名和调用大小写一致，是由于在类 UNIX 系统中，对大小写是敏感的（而 ThinkPHP 在调试模式下面，即使在 Windows 平台也会严格检查大小写）。
☑ 类名和文件名一致（包括上面说的大小写一致），如 UserAction 类的文件命名是 UserAction.class.php，InfoModel 类的文件名是 InfoModel.class.php。
☑ 函数的命名使用小写字母和下划线的方式，如 get_client_ip。
☑ Action 控制器类以 Action 为后缀，如 UserAction、InfoAction。
☑ 模型类以 Model 为后缀，如 UserModel、InfoModel。
☑ 方法的命名使用驼峰法，并且首字母小写，如 getUserName。
☑ 属性的命名使用驼峰法，并且首字母小写，如 tableName。
☑ 以双下划线（__）打头的函数或方法作为魔法方法，如__call 和__autoload。
☑ 常量以大写字母和下划线命名，如 HAS_ONE 和 MANY_TO_MANY。
☑ 配置参数以大写字母和下划线命名，如 HTML_CACHE_ON。
☑ 语言变量以大写字母和下划线命名，如 MY_LANG，以下划线开头的语言变量通常用于系统语言变量，如_CLASS_NOT_EXIST_。
☑ 数据表和字段采用小写加下划线方式命名，如 think_user 和 user_name。

说明

在 ThinkPHP 中有一个函数命名很特例，就是单字母大写函数，这类函数通常是某些操作的快捷定义，或者有特殊的作用。例如，ADSL 方法等，它们有着特殊的含义。另外一点，ThinkPHP 默认使用 UTF-8 编码，所以请确保程序文件采用 UTF-8 编码格式保存，并且去掉 BOM 信息头（去掉 BOM 头信息有很多方式，不同的编辑器都有设置方法，也可以用工具进行统一检测和处理）。

24.2.5　项目构建流程

ThinkPHP 具有项目目录自动创建功能，因此构建项目应用程序非常简单，用户只需定义好项目的入口文件，在第一次访问入口文件时，系统会自动根据在入口文件中所定义的目录路径，迅速创建好项目的相关目录结构。在完成项目目录结构的创建后，再进行其他工作，如图 24.7 所示展示了 ThinkPHP 创建项目的基本流程。

例 24.2　根据上述讲解的流程，创建一个名称为 TM 的项目，读取 db_database24 数据库中的数据。**（实例位置：光盘\TM\Instance\24\24.2）**

其操作步骤如下。

（1）创建 db_database24 数据库和 think_user 数据表。数据表结构如图 24.8 所示

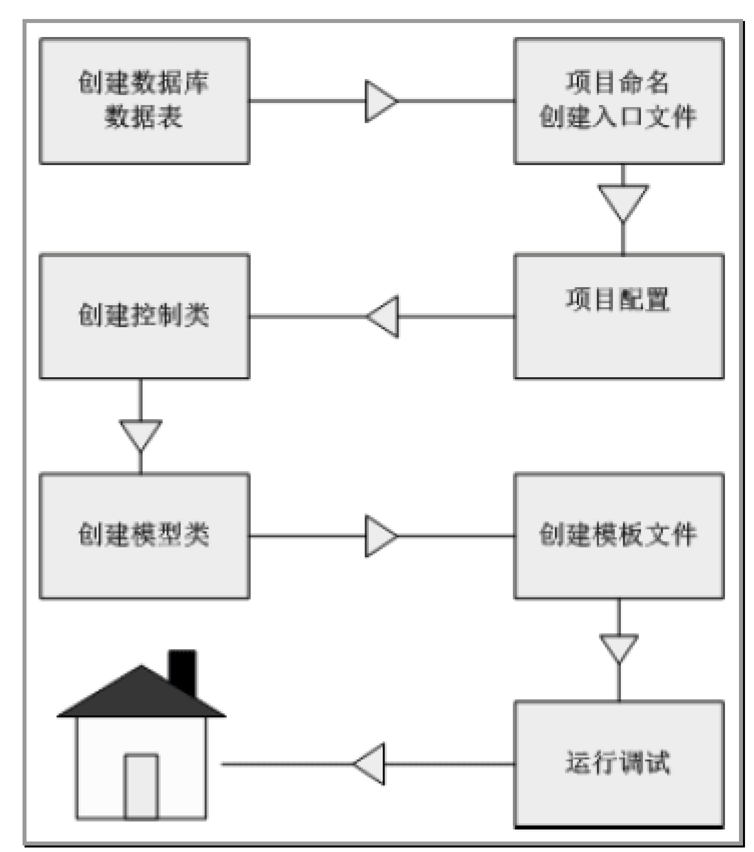


图 24.7　ThinkPHP 项目构建流程



服务器: localhost ▸ 数据库: db_database24 ▸ 表 : think_user

浏览　结构　SQL　搜索　插入　导出　Import　操作　清空　删除

	字段	类型	整理	属性	Null	默认	额外	操作
	id	int(10)			否		auto_increment	
	user	varchar(80)	utf8_unicode_ci		否			
	pass	varchar(80)	utf8_unicode_ci		否			
	address	varchar(80)	utf8_unicode_ci		否			

图 24.8　ThinkPHP 项目构建流程

（2）载入 ThinkPHP 系统文件，编辑入口文件 index.php，创建名称为 TM 的项目。index.php 的代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP/');                    //定义 ThinkPHP 框架路径
define('APP_NAME', 'TM');                                //定义项目名称和路径
define('APP_PATH', '.');                                 //定义项目名称和路径
require(THINK_PATH."/ThinkPHP.php");                     //加载框架入口文件
App::run();//实例化一个网站应用实例
?>
```

（3）自动生成的项目目录中已经创建了一个空的项目配置文件，位于项目的 Conf 目录下面，名称是 config.php。重新编辑此文件，完成数据库的配置。config.php 文件的代码如下：

```
<?php
return array(
	'APP_DEBUG' => true,                            //开启调试模式
	'DB_TYPE'=> 'mysql',                            //数据库类型
	'DB_HOST'=> 'localhost',                        //数据库服务器地址
	'DB_NAME'=>'db_database24',                     //数据库名称
	'DB_USER'=>'root',                              //数据库用户名
	'DB_PWD'=>'111',                                //数据库密码
	'DB_PORT'=>'3306',                              //数据库端口
	'DB_PREFIX'=>'think_',                          //数据表前缀
```

```
);
?>
```

（4）在项目的 Lib\Action 目录下，定位到自动生成的 IndexAction.class.php 文件，这是 ThinkPHP 的控制器，即 Index 模块。重新编辑控制器的 index 方法，查询指定数据表中的数据，并且完成数据的循环输出。其代码如下：

```
<?php
class IndexAction extends Action{
	public function index() {
		$db = new Model('user');                    //实例化模型类，参数数据表名称不包含前缀
		$select = $db->select();                     //查询数据
		$this->assign('select',$select);            //模板变量赋值
		$this->display();                            //输出模板
	}
}
?>
```

（5）在项目的 Tpl\default 目录下，创建 Index 目录，存储 Index 模块的模板文件 index.html，完成数据库中数据的循环输出。其代码如下：

```
<!--循环输出查询结果数据集-->
<volist name='select' id='user' >
ID:{$user.id}<br/>
用户名：  {$user.user}<br/>
地址：  {$user.address}<hr>
</volist>
```

（6）在 IE 浏览器中输入 http://127.0.0.1:82/TM/24/24.2/，其运行结果如图 24.9 所示。

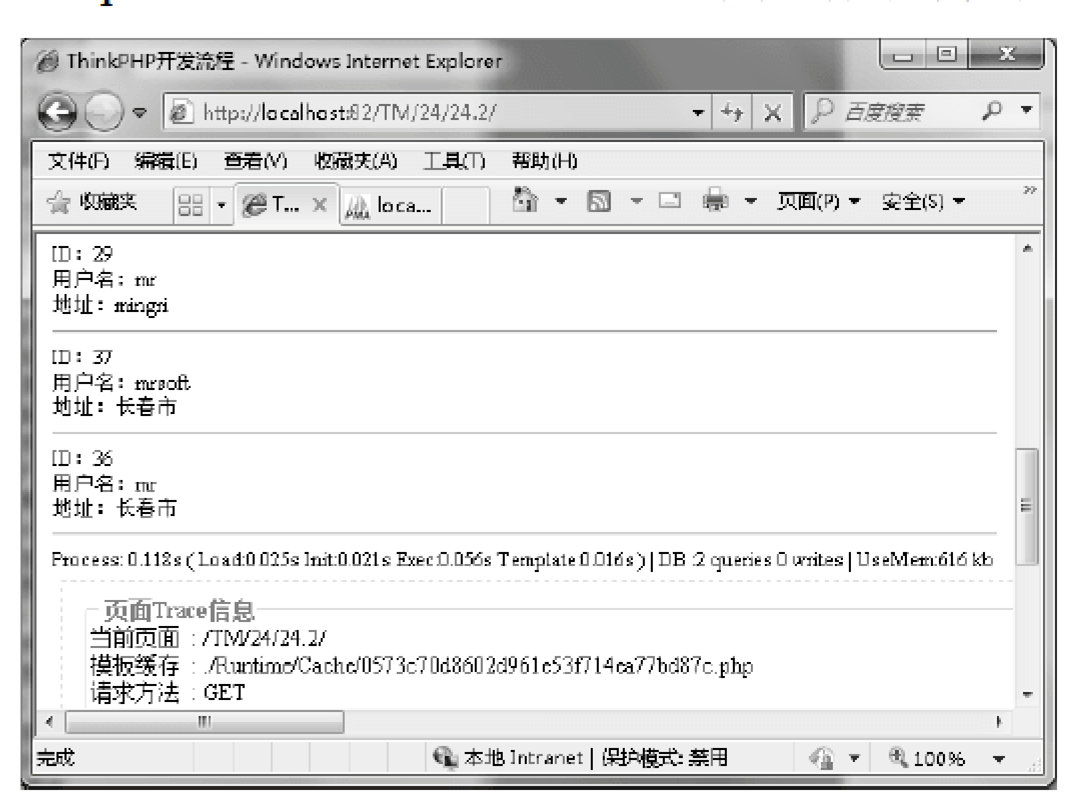


图 24.9　了解 ThinkPHP 项目构建流程

24.3　ThinkPHP 的配置

视频讲解：光盘\TM\Video\第 24 章\ThinkPHP 的配置.exe

配置文件是 ThinkPHP 框架程序得以运行的基础条件，框架的很多功能都需要在配置文件中配置后，才可以生效，包括 URL 路由功能、页面伪静态和静态化等。ThinkPHP 提供了灵活的全局配置功能，采用最有效率的 PHP 返回数组方式定义，支持惯例配置、项目配置、调试配置和模块配置，并且会自动生成配置缓存文件，无需重复解析。

ThinkPHP 在项目配置上面创造了自己独有的分层配置模式，其配置层次如图 24.10 所示。

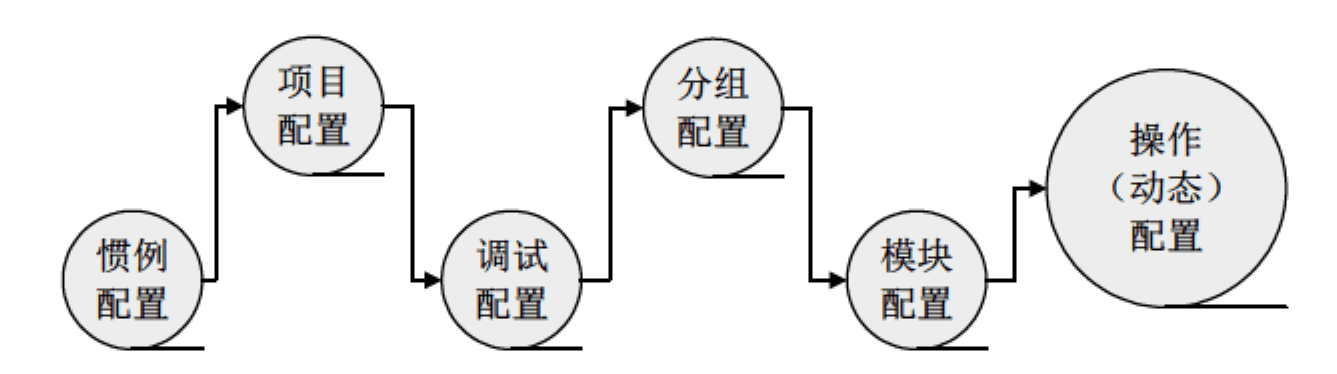


图 24.10　分层配置模式的顺序

以上是配置文件的加载顺序，但是因为后面的配置会覆盖之前的配置（在没有生效的前提下），所以优先顺序从右到左。系统的配置参数是通过静态变量全局存取的，存取方式非常得简单、高效。

24.3.1　配置格式

ThinkPHP 框架中所有配置文件的定义格式均采用返回 PHP 数组的方式。其格式如下：

```
<?php
return array(
	'APP_DEBUG' => true,
	'URL_MODEL' => 2,
	……//更多的配置参数
);
?>
```

说明 配置参数不区分大小写（因为无论使用大写或小写定义，都会转换成小写），但是习惯上保持大写定义的原则。另外，还可以在配置文件中使用二维数组来配置更多的信息。例如：

```
<?php
return array(
	'APP_DEBUG' => true,
	'USER_CONFIG' => array(
			'USER_AUTH' => true,
			'USER_TYPE' => 2,
	),
);
?>
```

系统目前最多支持二维数组的配置级别，每个项目配置文件除了定义 ThinkPHP 所需要的配置参数之外，开发人员还可以在里面添加项目需要的一些配置参数，用于自己的应用。项目配置文件默认存储于项目的 Conf 目录。例如，在例 24.2 中，连接数据库的配置文件存储于项目的 24.2\Conf\config.php 文件中。

技巧 项目配置指的是项目的全局配置，因为一个项目除了可以定义项目配置文件之外，还可以定义模块配置文件，用于针对某个特定的模块进行特殊的配置。它们的定义格式都是一致的，区别只是配置文件命名的不同。系统会自动在不同的阶段读取配置文件。

24.3.2　调试配置

如果启用调试模式的话，那么会导入框架默认的调试配置文件，默认的调试配置文件位于 Think\Common\debug.php 文件中，如果没有检测到项目的调试配置文件，就会直接使用默认的调试配置参

数。项目定义自身的调试配置文件，则会和默认的调试配置文件合并，也就是说，项目配置文件也只需要配置和默认调试配置不同的参数或者新增的参数。

调试配置文件也位于项目配置目录下面，文件名是 debug.php。通常情况下，调试配置文件里面可以进行一些开发模式所需要的配置。例如，配置额外的数据库连接用于调试、开启日志写入便于查找错误信息、开启页面 Trace 输出更多的调试信息等。系统默认的调试配置文件中设置如下内容。

- ☑ 开启日志记录。
- ☑ 关闭模板缓存。
- ☑ 记录 SQL 日志。
- ☑ 关闭字段缓存。
- ☑ 开启运行时间详细显示（包括内存、缓存情况）。
- ☑ 开启页面 Trace 信息显示。
- ☑ 严格检查文件大小写（即使是 Windows 平台）。

24.4 ThinkPHP 的控制器

视频讲解：光盘\TM\Video\第 24 章\ThinkPHP 的控制器.exe

24.4.1 控制器

ThinkPHP 的控制器就是模块类，通常位于项目的 Lib\Action 目录下面。类名就是模块名加上 Action 后缀，如 IndexAction 类表示 Index 模块。控制器类必须继承系统的 Action 基础类，这样才能确保使用 Action 类内置的方法。

例 24.3 对自动生成的项目目录中的控制器进行修改，使其输出自己编译的内容。**（实例位置：光盘\TM\Instance\24\24.3）**

其操作步骤如下。

（1）创建名称为 TM 的项目，将 ThinkPHP 核心类库存储于 TM 目录下。

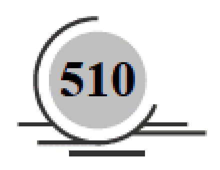
（2）编写入口文件 index.php，将其存储于 TM 目录下。index.php 文件代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP/');                    //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'TM');                                //定义项目名称
define('APP_PATH', '.');                                 //定义错误提示
require(THINK_PATH."/ThinkPHP.php");                     //加载框架入口文件
App::run();                                              //实例化一个网站应用实例
?>
```

（3）运行 index.php 文件，在 TM 目录下自动生成项目目录，其运行效果如图 24.11 所示。

（4）在默认生成的项目目录中，控制器 IndexAction 中输出的是 ThinkPHP 设置的内容，此时我们对这个内容进行修改，输出“明日科技欢迎您！”。修改后 24.3\Lib\Action\IndexAction.class.php 文件的代码如下：

```
<?php
class IndexAction extends Action{
    public function index(){
        header("Content-Type:text/html; charset=utf-8");            //设置编码格式
        echo "<div style='font-weight:normal;color:blue;float:left;width:345px;text-align:center;border:1px solid
```



```
silver;background:#E8EFFF;padding:8px;font-size:14px;font-family:Tahoma'>^_^ <span style='font-weight:bold;
color:red'> 明日科技欢迎您！</span></div>";                                    //输出内容
      }
}
?>
```

在对控制器的内容进行修改后，重新运行项目，在 IE 浏览器中输入 http://127.0.0.1:82/TM/24/24.3/，将输出如图 24.12 所示的效果。

^_^ Hello,欢迎使用ThinkPHP

图 24.11　自动创建项目目录

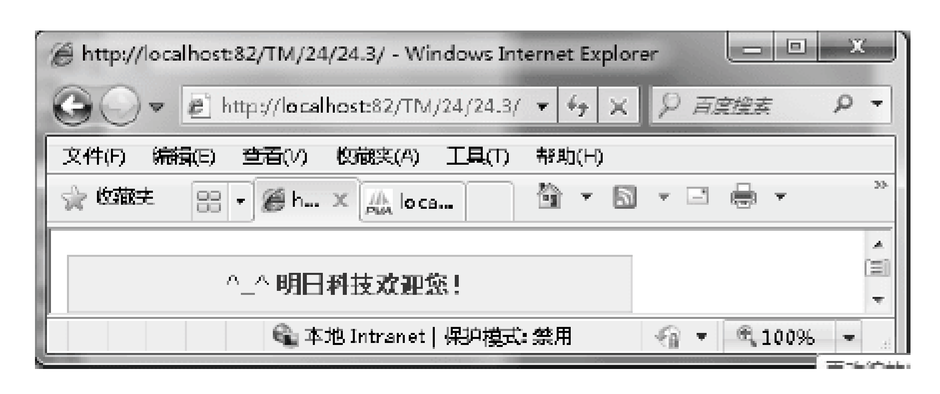


图 24.12　输出控制器中的内容

说明　每个模块的操作并非一定要定义操作方法，如果只是希望输出一个模板，既没有变量也没有任何的业务逻辑，那么只要按照规则定义好操作对应的模板文件即可，而不需要定义操作方法。例如，在 IndexAction 中如果没有定义 help 方法，但是存在对应的 Index/help.html 模板文件，那么 http://localhost:82/24/24.3/index.php/Index/help/依然可以正常运行，因为系统找不到 IndexAction 类的 help 方法，会自动定位到 Index 模块的模板目录中查找 help.html 模板文件，然后直接输出。代码中加粗的部分就是 Smarty 标签，大括号"{}"为标签的定界符，$title 和$content 为变量。

24.4.2　跨模块调用

在开发过程中经常会在当前模块调用其他模块的方法，这个时候就涉及到跨模块调用。下面通过 A 和 R 两个快捷方法完成跨模块调用。

```
$User = A("User");                          //实例化 UserAction 控制器对象
$User->insert();                            //调用 User 模块的 importUser 操作方法
```

这里的 A("User") 是一个快捷方法，与下面的代码等效：

```
import("@.Action.UserAction");
$User = new UserAction();
```

事实上，在这个例子里面还有比 A 方法更简单的调用方法，例如：

```
R("User","insert");                         //远程调用 UserAction 控制器的 insert 操作方法
```

上面只是在当前项目中调用，如果需要在多个项目之间调用方法，一样可以完成。

```
$User = A("User","Admin");                  //实例化 Admin 项目的 UserAction 控制器对象
$User->insert();                            //调用 Admin 项目 UserAction 控制器的 insert 操作方法
 R("User","insert","Admin");                //远程调用 Admin 项目的 UserAction 控制器的 insert 操作方法
```

例 24.4　应用跨模块调用的方法，在前台控制器中调用后台项目中的 insert 方法完成用户信息的添加操作。**（实例位置：光盘\TM\Instance\24\24.4）**

其操作步骤如下。

（1）创建 TM 项目根目录，在根目录下分别创建前台项目文件夹 Home、后台项目文件夹 Admin 和 Public 存储 CSS、图片和 JS 脚本等文件。

（2）在 TM 项目根目录下，编辑 index.php 前台入口文件和 admin.php 后台入口文件。其关键代码如下：

```
//Index.php 前台入口文件
<?php
```

```
define('THINK_PATH', '../ThinkPHP/');                //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', TM');                             //定义项目名称
define('APP_PATH', 'Home');                          //定义项目路径
require(THINK_PATH."/ThinkPHP.php");                 //加载框架入口文件
App::run();                                          //实例化一个网站应用实例
?>
//Admin.php 后台入口文件
<?php
define('THINK_PATH', '../ThinkPHP/');                //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'TM');                            //定义项目名称
define('APP_PATH', 'Admin');                         //定义项目路径
require(THINK_PATH."/ThinkPHP.php");                 //加载框架入口文件
App::run();                                          //实例化一个网站应用实例
?>
```

（3）在 IE 浏览器中运行前后台的入口文件，自动生成项目目录。

（4）定位到 Admin\Conf 目录下，编辑 config.php 文件，完成后台项目中数据库的配置。其代码如下：

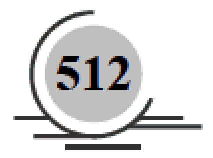
```
<?php
return array(
    'APP_DEBUG' => false,                            //关闭调试模式
    'DB_TYPE'=> 'mysql',                             //数据库类型
    'DB_HOST'=> 'localhost',                         //数据库服务器地址
    'DB_NAME'=>'db_database24',                      //数据库名称
    'DB_USER'=>'root',                               //数据库用户名
    'DB_PWD'=>'111',                                 //数据库密码
    'DB_PORT'=>'3306',                               //数据库端口
    'DB_PREFIX'=>'think_',                           //数据表前缀
);
?>
```

（5）定位到 Admin\Lib\Action 目录下，编写后台项目的控制器。首先创建 Index 模块，继承系统的 Action 基础类，定义 index 方法读取指定数据表中的数据，并且将查询结果赋给模板变量，最终指定模板页。IndexAction.class.php 的代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");     //设置页面编码格式
class IndexAction extends Action{
    public function index() {
        $db = new Model('user');                     //实例化模型类，参数数据表名称不包含前缀
        $select = $db->select();                     //查询数据
        $this->assign('select',$select);             //模板变量赋值
        $this->display();                            //输出模板
    }
}
?>
```

然后创建 User 模块，同样继承系统的 Action 基础类，定义 insert 方法，实例化模型类，将表单中提交的数据添加到指定的数据表中，添加成功后重定向到后台主页。UserAction.class.php 的代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");          //设置页面编码格式
class UserAction extends Action{                          //定义类，继承基础类
    public function insert() {                            //定义方法
        $ins = new Model('user');                         //实例化模型类，传递参数为没有前缀的数据表名称
        $ins->Create();                                   //创建数据对象
        $result = $ins->add();                            //写入数据库
        $this->redirect('Index/index','', 5,'页面跳转中');  //页面重定向
```



```
    }
}
?>
```

（6）定位到 Admin\Tpl\default 目录下，首先创建 Index 模块文件夹，编辑 index 操作的模板文件 index.html，循环输出模板变量传递的数据。index.html 模板文件的关键代码如下：

```
<volist name='select' id='user' >
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$user.id}</td>
    <td bgcolor="#FFFFFF"> {$user.user}</td>
    <td bgcolor="#FFFFFF"> {$user.address}</td>
  </tr>
  </volist>
```

然后创建 User 模块文件夹，编辑 insert 操作的模板文件 index.html，创建添加用户信息的表单。其关键代码如下：

```
<form method="post"   action="__URL__/insert" >
<table width="265" border="0" cellspacing="0" cellpadding="0">
            <tr>
              <td class="title" id="td">用户名：</td>
              <td><input name="user" type="text" size="15" /></td>
            </tr>
            <tr>
              <td class="title" id="td">密码：</td>
              <td><input name="pass" type="password" size="15" /></td>
            </tr>
            <tr>
              <td class="title" id="td">地址：</td>
              <td><input name="address" type="text" size="20" /></td>
            </tr>
          </table>
<input type="image" name="imageField" id="imageField" src="__ROOT__/Public/images/66_05.gif" />
</form>
```

（7）定位到 Home\Conf 目录，编辑前台项目的数据库配置文件 config.php。

（8）定位到 Home\Lib\Action 目录，创建前台项目控制器 Index。定义 index 方法查询数据库中的用户信息，并且将查询结果赋给模板变量；定义 insert 方法，通过 R 快捷方式调用 Admin 项目 UserAction 控制器的 insert 操作方法完成数据的添加操作。IndexAction.class.php 的代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");     //设置页面编码格式
class IndexAction extends Action{
    public function index() {
        $db = new Model('user');                     //实例化模型类，参数数据表名称不包含前缀
        $select = $db->select();                     //查询数据
        $this->assign('select',$select);             //模板变量赋值
        $this->display();                            //输出模板
    }
    public function insert() {
        $ins=R("User","insert","Admin");             //远程调用 Admin 项目 UserAction 控制器的 insert 操作方法
        $ins->Create();                              //创建数据对象
        $result = $ins->add();                       //写入数据库
    }
}
?>
```

（9）定位到 Home\Tpl\default 目录下，创建 Index 模块文件夹，编辑 index 操作的模板文件 index.html。

在 index.html 模板文件中，创建表单提交用户注册信息，循环输出模板变量传递的数据。

（10）在 IE 浏览器中输入 http://127.0.0.1:82/TM/24/24.4/，其运行效果如图 24.13 所示。



图 24.13　跨模块调用完成用户注册

技巧 Action 类的 redirect 方法可以实现页面的重定向功能。redirect 方法的定义规则如下（方括号内参数根据实际应用决定）:

redirect('[项目://][路由@][分组名-模块/]操作？ 参数 1=值 1[&参数 N=值 N]')

或者用数组的方式传入参数:

redirect('[项目://][路由@][分组名-模块/]操作',array('参数 1'=>'值 1' [,'参数 N'=>'值 N']))

如果不定义项目和模块的话，就表示当前项目和模块名称。例如:

```
$this->redirect('Index/index','', 5,'页面跳转中');                    //页面重定向
```

停留 5 秒后跳转到 Index 模块的 index 操作方法，并且显示页面跳转中字样，重定向后会改变当前的 URL 地址。

24.5　ThinkPHP 的模型

视频讲解：光盘\TM\Video\第 24 章\ThinkPHP 的模型.exe

顾名思义，模型就是按照某一个形状进行操作的代名词。模型的主要作用是，封装数据库的相关逻辑。也就是说，每执行一次数据库操作，都要遵循定义的数据模型规则来完成。

24.5.1　模型的命名

在定义模型时，ThinkPHP 要求数据库的表名和模型类的命名遵循一定的规范，首先数据库的表名和字段全部采用小写形式，模型类的命名规则是除去表前缀的数据表名称，并且首字母大写，然后加上模型类的后缀定义。

例如，UserModel 表示 User 数据对象，（假设数据库的前缀定义是 think_）其对应的数据表应该是 think_user；UserTypeModel 对应的数据表是 think_user_type。

如果你的规则和系统的约定不符合，那么需要设置 Model 类的 tableName 属性。在 ThinkPHP 的模型里面，有两个数据表名称的定义。

（1）tableName 不包含表前后缀的数据表名称，一般情况下默认和模型名称相同，只有当表名和当前的模型类的名称不同的时候才需要定义。例如，在数据库里面有一个 think_categories 表，而定义的模型类名称是 CategoryModel，按照系统的约定，这个模型的名称是 Category，对应的数据表名称应该是

think_category（全部小写），但是现在的数据表名称是 think_categories，因此就需要设置 tableName 属性来改变默认的规则（假设已经在配置文件里面定义了 DB_PREFIX 为 think_）。

```
protected $tableName = 'categories';
```


这个属性的定义不需要加表的前缀 think_。

（2）trueTableName 包含前后缀的数据表名称，也就是数据库中的实际表名，该名称无需设置，只有当上面的规则都不适用的情况或者特殊情况下才需要设置。例如，数据库中有一个表（top_depts）的前缀和其他表前缀不同，不是 think_ 而是 top_，这个时候需要定义 trueTableName 属性。

```
protected $trueTableName = 'top_depts';
```


trueTableName 需要完整的表名定义。

除了数据表的定义外，还可以对数据库进行定义。

dbName 定义模型当前对应的数据库名称，只有当前的模型类对应的数据库名称和配置文件不同时才需要定义，例如。

```
protected $dbName = 'top';
```

24.5.2　实例化模型

在 ThinkPHP 2.0 版本中，无需进行任何模型定义（只有在需要封装单独的业务逻辑时，模型类才是必须被定义的），可以直接进行模型的实例化操作。根据不同的模型定义，实例化模型的方法也有所不同。下面来分析一下什么情况下使用什么方法。

1．实例化基础模型（Model）类

在没有定义任何模型时，可以使用下面的方法实例化一个模型类来进行操作。

```
$User = new Model('User');
$User->select();                                    //进行其他的数据操作
```

或者使用 M 快捷方法进行实例化，其效果是相同的。

```
$User = M('User');
$User->select();                                    //进行其他的数据操作
```

这种方法最简单、高效，因为不需要定义任何的模型类，所以支持跨项目调用。缺点也是因为没有自定义的模型类，因此无法写入相关的业务逻辑，只能完成基本的 CURD 操作。在例 24.2 和例 24.4 中采用的都是实例化基础模型类，对数据库中数据进行读取和添加操作。

2．实例化其他模型类

第一种方式实例化因为没有模型类的定义，因此很难封装一些额外的逻辑方法，不过大多数情况下，也许只是需要扩展一些通用的逻辑，那么就可以尝试下面一种方法。

M 方法默认是实例化 Model 类，如果需要实例化其他模型类，可以使用：

```
$User = M('User', 'CommonModel');
```

上面的方法等效于：

```
$User = new CommonModel('User');
```

因为系统的模型类都能够自动加载，因此不需要在实例化之前手动进行类库导入操作。模型类 commonModel 必须继承 Model，如果没有定义别名导入的话，需要放在项目 Model 下。我们可以在

CommonModel 类里面定义一些通用的逻辑方法，就可以省去为每个数据表定义具体的模型类，如果项目的数据表超过 100 个，而且大多数都是执行基本的 CURD 操作，只是个别模型有一些复杂的业务逻辑需要封装，那么第一种方式和第二种方式的结合是一个不错的选择。

3．实例化用户定义的模型（×××Model）类

这种情况是使用最多的，一个项目不可避免地需要定义自身的业务逻辑实现，就需要针对每个数据表定义一个模型类，如 UserModel、InfoModel 等。

定义的模型类通常都是放到项目的 Lib\Model 目录下面。例如：

```
class UserModel extends Model{
    Public function myfun(){
        //添加自己的业务逻辑
    }
}
```

其实模型类还可以继承一个用户自定义的公共模型类，而不是只能继承 Model 类。要实例化自定义模型类，可以使用 D 快捷方法，其效果是相同的。

```
$User = D('User');
$User->select();                                    //进行其他的数据操作
```

D 方法可以自动检测模型类，不存在时系统会抛出异常，同时对于已实例化过的模型，不会重复去实例化。默认的 D 方法只能支持调用当前项目的模型，如果需要跨项目调用，需要使用：

```
$User = D('User', 'Admin');                         //实例化 Admin 项目下面的 User 模型
$User->select();
```

如果启用模块分组功能，还可以使用：

```
$User = D('Admin.User');
```

4．实例化空模型类

如果仅仅是使用原生 SQL 查询的话，不需要使用额外的模型类，实例化一个空模型类即可进行操作。例如：

```
$Model = new Model();
//或者使用 M 快捷方法实例化是等效的
//$Model = M();
$Model->query('SELECT * FROM think_user where status=1');
```

空模型类也支持跨项目调用。

例 24.5　通过 M 方法实例化 Model 类，完成数据库中用户信息和类别信息的输出。（**实例位置：光盘\TM\Instance\24\24.5**）

其关键操作步骤如下。

（1）创建 TM 项目根目录，在根目录下创建项目文件夹 App 和 Public 存储 CSS、图片和 JS 脚本等文件。

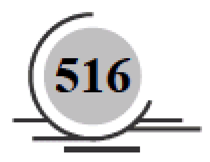
（2）在 TM 项目根目录下，编辑 index.php 入口文件。其关键代码如下：

```
<?php
define('THINK_PATH', '../ThinkPHP/');              //定义 ThinkPHP 框架路径（相对于入口文件）
define('APP_NAME', 'TM');                          //定义项目名称
define('APP_PATH', 'App');                         //定义项目路径
require(THINK_PATH."/ThinkPHP.php");               //加载框架入口文件
App::run();                                        //实例化一个网站应用实例
?>
```

（3）在 IE 浏览器中运行入口文件，自动生成项目目录。

（4）定位到 App\Conf 目录下，编辑 config.php 文件，完成项目中数据库的配置。其代码如下：

```
<?php
```



```
return array(
        'APP_DEBUG' => false,                  //关闭调试模式
        'DB_TYPE'=> 'mysql',                   //数据库类型
        'DB_HOST'=> 'localhost',               //数据库服务器地址
        'DB_NAME'=>'db_database24',            //数据库名称
        'DB_USER'=>'root',                     //数据库用户名
        'DB_PWD'=>'111',                       //数据库密码
        'DB_PORT'=>'3306',                     //数据库端口
        'DB_PREFIX'=>'think_',                 //数据表前缀
);
?>
```

（5）定位到 App\Lib\Action 目录下，编写项目的控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，通过 M 方法实例化模型类，读取 think_user 数据表中的数据，并且将查询结果赋给模板变量，指定模板页；定义 type()方法，通过 M 方法实例化模型类，读取类型数据表 think_type 中的数据，同样将查询结果赋给模板变量，指定模板页。IndexAction.class.php 的代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");       //设置页面编码格式
class IndexAction extends Action{
        public function index(){
                $db = M('User');                       //实例化模型类，参数数据表名称不包含前缀
                $select = $db->select();               //查询数据
                $this->assign('select',$select);       //模板变量赋值
                $this->display();                      //指定模板页
        }
        public function type(){
                $dba = M('Type');                      //实例化模型类，参数数据表名称不包含前缀
                $select = $dba->select();              //查询数据
                $this->assign('select',$select);       //模板变量赋值
                $this->display('type');                //指定模板页
        }
}
?>
```

（6）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。首先，编辑 index 操作的模板文件 index.html，循环输出模板变量传递的数据。其关键代码如下：

```
<volist name='select' id='user' >
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$user.id}</td>
    <td bgcolor="#FFFFFF"> {$user.user}</td>
    <td bgcolor="#FFFFFF"> {$user.address}</td>
  </tr>
</volist>
```

然后编辑 type.html 模板文件，循环输出类型数据表中的数据。其关键代码如下：

```
<volist name='select' id='type' >
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$type.id}</td>
    <td bgcolor="#FFFFFF"> {$type.typename}</td>
    <td bgcolor="#FFFFFF"> {$type.dates}</td>
  </tr>
</volist>
```

（7）在 IE 浏览器中输入 http://127.0.0.1:82/TM/24/24.5/，其运行效果如图 24.14 所示。在 IE 浏览器中输入 http://127.0.0.1:82/TM/24/24.5/index.php/Index/type，其运行效果如图 24.15 所示。

用户信息		
ID	名称	地址
1	mr	长春市
2	mrsoft	四平市
3	Tsoft	长春市

图 24.14　输出用户信息

类别输出		
ID	类别名称	添加时间
1	PHP	2011-05-16
2	JAVA	2011-05-16
3	C#	2011-05-16
4	C++	2011-05-16

图 24.15　输出类别信息

说明 在 Model 类里面根本没有定义任何 User 表、Type 表的字段信息，但是系统是如何做到属性对应数据表的字段呢？这是因为 ThinkPHP 可以在运行时自动获取数据表的字段信息（确切地说，是在第一次运行时，将其存储于缓存文件，以后会永久缓存字段信息，除非设置不缓存或者删除），包括数据表的主键字段和是否自动增长等，如果需要显式获取当前数据表的字段信息，可以使用模型类的 getDbFields 方法来获取。如果在开发过程中修改了数据表的字段信息，需要清空 Data/_fields 目录下面的缓存文件，让系统重新获取更新的数据表字段信息。

24.5.3　属性访问

ThinkPHP 利用 PHP 5 的魔术方法机制来实现属性的直接访问。这也是最常用的访问方式，通过数据对象访问，例如：

```
<?php
$User = new Model('User');
$User->find(1);
echo $User->name;                                    //获取 name 属性的值
$User->name = 'ThinkPHP';                            //设置 name 属性的值
?>
```

还有一种属性的操作方式是通过返回数组的方式。例如：

```
<?php
$Type = D('Type');                                   //注意这里返回的 type 数据是一个数组
$type = $Type->find(1);
echo $type['name'];                                  //获取 type 属性的值
$type['name'] = 'ThinkPHP';                          //设置 type 属性的值
?>
```

24.5.4　连接数据库

ThinkPHP 内置抽象数据库访问层，把不同的数据库操作封装起来，只需使用公共的 Db 类进行操作，而无需针对不同的数据库写不同的操作代码，Db 类会自动调用相应的数据库适配器来处理。目前的数据库包括 Mysql、MsSQL、PgSQL、Sqlite、Oracle、Ibase 以及 PDO 的支持，如果应用需要使用数据库，必须配置数据库连接信息，数据库的配置文件有多种定义方式。

（1）在项目配置文件里面定义，在前面的实例中已经见识过了。其代码如下：

```
<?php
return array(
      'APP_DEBUG' => false,                          //关闭调试模式
      'DB_TYPE'=> 'mysql',                           //数据库类型
      'DB_HOST'=> 'localhost',                       //数据库服务器地址
      'DB_NAME'=>'db_database24',                    //数据库名称
      'DB_USER'=>'root',                             //数据库用户名
      'DB_PWD'=>'111',                               //数据库密码
```

```
        'DB_PORT'=>'3306',                                              //数据库端口
        'DB_PREFIX'=>'think_',                                          //数据表前缀
);
?>
```

系统推荐使用该种方式，因为一般一个项目的数据库访问配置是相同的。该方法系统在连接数据库时会自动获取，无需手动连接。

说明 可以对每个项目定义不同的数据库连接信息，还可以在调试配置文件里面定义调试数据库的配置信息，如果在项目配置文件和调试模式配置文件里面同时定义了数据库连接信息，那么在调试模式下面后者生效，部署模式下面前者生效。

（2）使用 DSN 方式在初始化 Db 类时传参数，代码如下：

```
$db_dsn="mysql://root:111@127.0.0.1:3306/db_database24";        //定义 DSN
$db = new Db();                                                 //执行类的实例化
$conn=$db->getInstance($db_dsn);                                //连接数据库，返回数据库驱动类
```

该方式主要用于在控制器里面自己手动连接数据库的情况，或者用于创建多个数据库连接。

（3）第三种为使用数组传参数，代码如下：

```
$dsn = array(
        'dbms' => 'mysql',
        'username' => 'username',
        'password' => 'password',
        'hostname' => 'localhost',
        'hostport' => '3306',
        'database' => 'dbname'
);
$db = new Db();
$conn=$db->getInstance($dsn);                                   //连接数据库，返回数据库驱动类
```

该方式用于手动连接数据库或者创建多个数据库连接。

例 24.6 通过 DNS 方式和数组传参的方式完成与数据库的连接，并且输出数据库中的数据。**（实例位置：光盘\TM\Instance\24\24.6）**

其关键操作步骤如下。

本示例是例 24.5 的延伸，仍然输出数据库中用户和类别表中的数据，只是对其连接数据库的方法进行了修改。

（1）删除 App\Conf 目录下的配置文件 config.php。

（2）在 App\Lib\Action\Index 目录下，修改控制器 Index。在 index()方法中，应用 DNS 方式完成与数据库的连接，并且查询 think_user 表中的数据。其关键代码如下：

```
        public function index(){
                $db_dsn="mysql://root:111@127.0.0.1:3306/db_database24";  //定义 DSN
                $db = new Db();                                           //执行类的实例化
                $conn=$db->getInstance($db_dsn);                          //连接数据库，返回数据库驱动类
                $select=$conn->query('select * from think_user');         //执行查询语句
                $this->assign('select',$select);                          //模板变量赋值
                $this->display();                                         //指定模板页
        }
```

（3）在 type()方法中，应用数组传递参数，完成数据库的连接操作，并且查询 think_type 表中的数据。其关键代码如下：

```
public function type(){
        $dsn = array(
```

```
                'dbms'          => 'mysql',
                'username' => 'root',
                'password' => '111',
                'hostname' => 'localhost',
                'hostport' => '3306',
                'database' => 'db_database24'
        );
        $db = new Db();
        $conn=$db->getInstance($dsn);                              //连接数据库，返回数据库驱动类
        $select=$conn->query('select * from think_type');          //执行查询语句
        $this->assign('select',$select);                           //模板变量赋值
        $this->display('type');                                    //指定模板页
    }
```

上述是在例 24.5 中所做的修改，至于其他步骤与例 24.5 相同，这里不再赘述了。其运行结果也与例 24.5 相同。

（4）在模型类里面定义参数，连接数据库，代码如下：

```
protected $connection = array(
    'dbms' => 'mysql',
    'username' => 'username',
    'password' => 'password',
    'hostname' => 'localhost',
    'hostport' => '3306',
    'database' => 'dbname'
);
//或者使用下面的方式定义
protected $connection = "mysql://username:password@localhost:3306/DbName";
```

如果在某个模型类里面定义了 connection 属性，则在实例化模型对象时，会使用该数据库连接信息进行数据库连接。通常用于某些数据表位于当前数据库连接之外的其他数据库。

说明 ThinkPHP 并不是在一开始就会连接数据库，而是在有数据查询操作时才会去连接数据库。特殊的情况是，在系统第一次操作模型时，框架会自动连接数据库获取相关模型类的数据字段信息，并缓存下来。

（5）使用 PDO 方式连接数据库。这里在项目配置文件中，应用 PDO 连接数据库。其定义的数组内容如下：

```
return array(
    'DB_TYPE'=> 'pdo',
    //注意 DSN 的配置针对不同的数据库有所区别
    'DB_DSN'=> 'mysql:host=localhost;dbname=db_database24',
    'DB_USER'=>'root',
    'DB_PWD'=>'111',
    'DB_PREFIX'=>'think_',
    //其他项目配置参数
    'APP_DEBUG' => false,                          //关闭调试模式
);
```

注意 在使用 PDO 方式时，要注意检查你的 PHP 环境是否开启相关的 PDO 模块。同时还要确保你的 ThinkPHP 核心包中包含 DbPdo.class.php 文件。另外，还要注意参数 DB_DSN 仅对 PDO 方式连接才有效。



例 24.7　在项目配置文件中，以 PDO 方式连接数据库，并且输出数据库中的数据。**（实例位置：光盘\TM\24\24.7）**

我们知道在例 24.5 中数据库的连接方法定义到配置文件 config.php 中，而应用 PDO 连接 MySQL 数据库仍然需要在配置文件中进行操作，那么我们只需对例 24.5 中的 config.php 文件进行修改，就完成了一个新的实例，应用 PDO 连接 MySQL 数据，并且输出查询结果。其修改后的 config.php 文件的代码如下：

```
<?php
return array(
	'DB_TYPE'=> 'pdo',
	//注意 DSN 的配置针对不同的数据库有所区别
	'DB_DSN'=> 'mysql:host=localhost;dbname=db_database24',
	'DB_USER'=>'root',
	'DB_PWD'=>'111',
	'DB_PREFIX'=>'think_',
	…//其他项目配置参数
	'APP_DEBUG' => false,					//关闭调试模式
);
?>
```

本例的运行结果与 24.5 相同，这里不再赘述。

24.5.5　创建数据

ThinkPHP 可以自动根据表单数据创建数据对象，这个优势在一个数据表的字段非常之多的情况下尤其明显。例如，在 User 控制器中，定义 insert()方法，首先实例化模型类，然后调用 create()方法根据表单提交的 POST 数据创建数据对象，最后调用 add()方法把创建的数据对象写入数据库。其关键代码如下：

```
class UserAction extends Action{				//定义类，继承基础类
	public function insert() {				//定义方法
		$ins = new Model('user');		//实例化模型类，传递参数为没有前缀的数据表名称
		$ins->create();				//创建数据对象
		$result = $ins->add();			//写入数据库
		$this->redirect('Index/index','', 5,'页面跳转中');	//页面重定向
	}
}
```

短短的 3 行代码，完成数据的添加操作。其具体应用可以参考例 24.4。

其中的 create()方法还支持其他方式提交的数据对象。例如，以数组形式提交数据，从其他的数据对象中获取的数据等。其关键代码如下：

```
//数组形式提交数据
$data['user'] = 'mrsoft';
$data['address'] = '长春市';
$User->create($data);
//从 User 数据对象创建新的 Member 数据对象
$User = M("User");					//实例化 User 对象
$User->find(1);						//读取数据
$Member = M("Member");				//创建 Member 对象
$Member->create($User);
```

create()方法在创建数据对象的同时，还实现了一些非常有意义的功能，包括支持多种数据源、数据自动验证、字段类型检查和数据自动完成等。

create()方法创建的数据对象是保存在内存中，并没有实际写入到数据库中，直到使用 add()或者 save()方法，才真正将数据添加到数据库中。如果只是想简单创建一个数据对象，那么可以使用 data()方法。例如，

实例化 User 模型，通过 data()和 add()方法将数据添加到数据库中。其关键代码如下：

```
$User = M('User');                                          //实例化 User 模型
//创建数据后写入到数据库
$data['user'] = 'mrsoft';
$data['address'] = '长春市';
$User->data($data)->add();                                  //执行数据对象的创建
```

注意 使用 data()方法创建的数据对象不会进行自动验证和过滤操作，需要自行处理。但在进行 add 或者 save 操作时，数据表中不存在的字段以及非法的数据类型（例如对象、数组等非标量数据）是会自动过滤的，不用担心非数据表字段的写入导致 SQL 错误的问题。

24.5.6 连贯操作

ThinkPHP 2.0 版本全面启用模型类的连贯操作方法，可以有效地提高数据存取的代码清晰度和开发效率。例如，查询一个 User 表的满足状态为 1 的前 5 条记录，并按照用户的 ID 排序。其关键代码如下：

```
$User->where('status=1')->order('id')->limit(5)->select();
```

在连贯操作中，select()方法必须放到最后一个，其他的连贯操作方法调用顺序没有先后。如果不习惯使用连贯操作，那么新版还支持直接使用参数进行查询的方式。例如上面的代码可以改写为：

```
$User->select(array('order'=>'id', 'where'=>'status=1', 'limit'=>'5'));
```

使用数组参数方式的话，索引的名称就是连贯操作的方法名称。其实不仅仅是查询方法可以使用连贯操作，包括 add、 save、delete 等方法都可以使用。例如：

```
$User->where('id=1')->field('id,user,address')->find();
$User->where('status=1 and id=1')->delete();
```

下面对连贯操作的方法进行一下总结（更多的用法将在 CURD 操作的过程中详细描述），如表 24.3 所示。

表 24.3 连贯操作方法总结

方 法 名	描 述
where	用于查询或者更新条件的定义。参数支持字符串、数组和对象
table	定义要操作的数据表名称。可以动态改变当前操作的数据表名称，需要写数据表的全名，包含前缀，可以使用别名，例如：$Model->table('think_user user')->where('status>1')->select(); table()方法的参数支持字符串和数组，数组方式的用法： $Model->table(array('think_user'=>'user','think_group'=>'group'))->where('status>1')->select(); 使用数组方式定义的优势是可以避免因为表名和关键字冲突而出错的情况。如果不定义 table()方法，默认会自动获取当前模型对应或者定义的数据表
data	数据对象赋值。可以用于新增或者保存数据之前的数据对象赋值，例如： $Model->data($data)->add(); $Model->data($data)->where('id=3')->save(); Data 方法的参数支持对象和数组，如果是对象会自动转换成数组。如果不定义 data()方法赋值，也可以使用 create()方法或者手动给数据对象赋值的方式
field	定义要查询的字段。参数支持字符串和数组，例如： $Model->field('id,nickname as name')->select(); $Model->field(array('id','nickname'=>'name'))->select(); 如果不使用 field()方法指定字段的话，默认和使用 field('*')等效

续表

方　法　名	描　　述
order	对结果进行排序。例如，order('id desc') 排序方法支持对多个字段的排序，例如，order('status desc,id asc') order()方法的参数支持字符串和数组，数组的用法如下： order(array('status'=>'desc','id'))
limit	结果限制。在 ThinkPHP 中，无论操作的是 MySQL、MS SQL Server，还是 Oracle 数据库，其 limit 的方法是统一的，即 limit('offset,length')。例如，limit('1,10')，获取从第一条记录开始的 10 条记录。 注意，limit('10') 与 limit('0,10')是等效的
page	查询分页。属于新增特性，可以更加快速地进行分页查询。Page()方法的用法和 limit()方法类似，格式为：Page('page[,listRows]') Page 表示当前的页数，listRows 表示每页显示的记录数。例如，Page('2,10')，表示每页显示 10 条记录，获取第 2 页的数据。 listRow 如果不写的话，会读取 limit('length') 的值，例如，limit(25)->page(3);，表示每页显示 25 条记录，获取第 3 页的数据。如果 limit 也没有设置的话，则默认为每页显示 30 条记录
group	查询 Group 支持。例如，group('user_id')，group()方法的参数只支持字符串



有关上述方法的应用，将在后面的 CURD 操作中体现，这里不再一一举例。

24.5.7　CURD 操作

ThinkPHP 提供了灵活和方便的数据操作方法，CURD（创建、更新、读取和删除）是 4 个最基本的数据库操作。CURD 操作通常与连贯操作配合使用。下面将对各种操作的使用方法进行分析（在执行类的实例化操作时，统一使用 M 方法）。

1．创建操作

在 ThinkPHP 中使用 add()方法完成数据的添加操作。其使用方法如下：

```
$User = M("User");                                    //实例化 User 对象
$data['name'] = 'ThinkPHP';
$data['email'] = 'ThinkPHP@gmail.com';
$User->add($data);
```

或者使用 data()方法进行连贯操作。其代码如下：

```
$User->data($data)->add();
```

如果在 add 之前已经创建数据对象的话（如使用了 create()或者 data()方法），add()方法就不需要再传入数据了。

2．读取数据

在 ThinkPHP 中读取数据的方式很多，通常分为读取某个字段的值、读取数据和读取数据集。读取字段的值使用 getField()方法，读取数据使用 find()方法，读取数据集使用 select()方法。

getField()方法读取某个字段的值，如果传入多个字段的话，可以返回一个关联数组。返回的 list 是一个数组，键名是用户的 id，键值是用户的昵称 nickname。例如，获取 ID 为 3 的用户的昵称和获取所有用户的 ID 和昵称列表。

```
$User = M("User");                                    //实例化 User 对象
$nickname = $User->where('id=3')->getField('nickname');    //获取 ID 为 3 的用户的昵称
```



```
$list = $User->getField('id,nickname');                    //获取所有用户的 ID 和昵称列表
```

select()方法的返回值是一个二维数组，如果没有查询到任何结果的话，也是返回一个空的数组。配合上面提到的连贯操作方法可以完成复杂的数据查询。例如，查找 status 值为 1 的用户数据并以创建时间排序返回 10 条数据。

```
$User = M("User");                                         //实例化 User 对象
$list = $User->where('status=1')->order('create_time')->limit(10)->select();
```

find()方法与 select()类似，select 可用的所有连贯操作方法也都可以用于 find()方法，区别在于 find()方法最多只会返回一条记录，因此 limit()方法对于 find 查询操作是无效的。例如，查找 status 值为 1、name 值为 think 的用户数据。

```
$User = M("User");                                         //实例化 User 对象
$User->where('status=1 and name="think" ')->find();
```



注意 即使满足条件的数据不止一条，find()方法也只会返回第一条记录。

例 24.8 通过 add()方法向数据库中添加数据，然后，查询数据表中用户名等于 mr 的记录，按照降幂排列，循环输出 3 条记录。**（实例位置：光盘\TM\Instance\24\24.8）**

这里在讲解实例的实现步骤过程中，省略了项目目录的创建、入口文件的编写和配置文件的设置，其具体步骤可以参考例 24.4 或者例 24.5。这里将直接讲解在控制器中如何完成数据的添加和查询操作。

其关键操作步骤如下。

（1）定位到 24.8\App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，通过 M 方法实例化模型类，应用连贯操作中的 where()、order()、limit()和 select()方法读取 think_user 数据表中的数据，并且将查询结果赋给模板变量，指定模板页；定义 insert()方法，通过 M 方法实例化模型类，应用 add()方法向指定的数据表中添加数据。IndexAction.class.php 的代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");           //设置页面编码格式
class IndexAction extends Action{
    public function index(){
        $db = M('User');                                   //实例化模型类，参数数据表名称不包含前缀
        $select = $db->where('user="mr"')->order('id desc')->limit(3)->select();    //执行查询语句
        $this->assign('select',$select);                   //模板变量赋值
        $this->display();                                  //指定模板页
    }
    public function insert(){
        $dba = M('User');                                  //实例化模型类，参数数据表名称不包含前缀
        $data['user'] = 'mr';
        $data['pass'] = md5('mrsoft');
        $data['address'] = '长春市';
        $result=$dba->add($data);                          //执行添加数据
        if($result){
            $this->redirect('Index/index','', 2,'页面跳转中');    //页面重定向
        }
    }
}
?>
```

（2）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，应用 ThinkPHP 内置模板引擎中的 foreach 标签循环输出模板变量传递的数据；创建添加数据的表单，将数据提交到控制器的 insert()方法中进行处理。其关键代码如下：

```
<foreach name='select' item='user' >
```



```
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$user.id}</td>
    <td bgcolor="#FFFFFF"> {$user.user}</td>
    <td bgcolor="#FFFFFF"> {$user.address}</td>
  </tr>
  </foreach>
<form id="form1" name="form1" method="post" action="__URL__/insert">
<input type="submit" name="button" id="button" value="数据添加" />
</form>
```

其运行效果如图 24.16 所示。

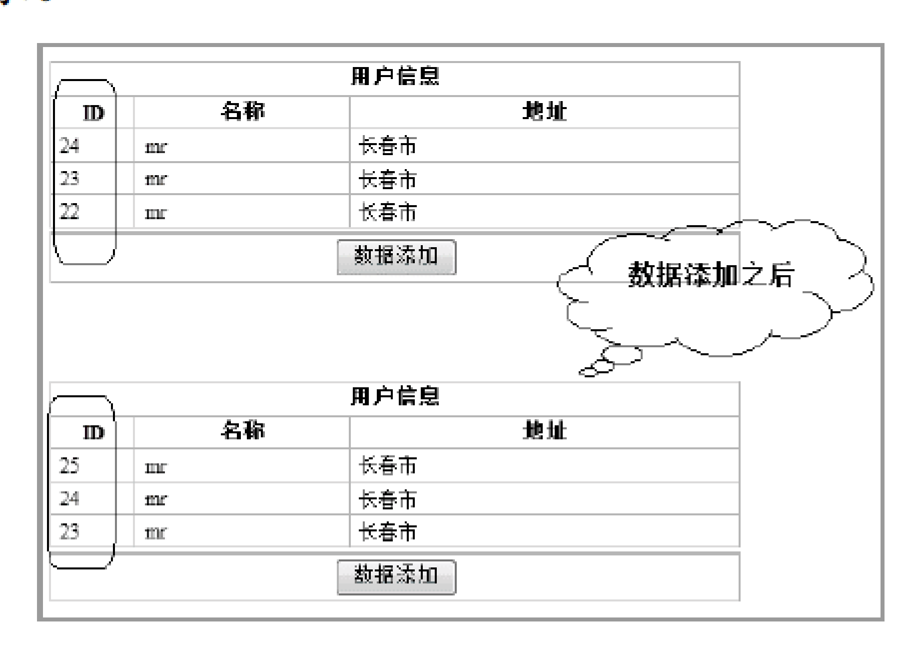


图 24.16 数据添加、查询后的运行结果

3. 更新数据

在 ThinkPHP 中使用 save()方法更新数据库，并且也支持连贯操作的使用。例如，更新数据表中 name 和 email 字段的值。其代码如下：

```
$User = M("User");                                  //实例化 User 对象
$data['name'] = 'ThinkPHP';                         //要修改的数据对象属性赋值
$data['email'] = 'ThinkPHP@gmail.com';
$User->where('id=5')->save($data);                  //根据条件保存修改的数据
```

save()方法在执行更新数据的操作时，如果没有设置任何更新条件，且数据对象本身也不包含主键字段，那么 save 方法不会更新任何数据库的记录。

第二种方法通过 data()方法创建要更新的数据对象，然后通过 save()方法进行保存。例如：

```
$User = M("User");                                  //实例化 User 对象
$data['name'] = 'ThinkPHP';                         //要修改的数据对象属性赋值
$data['email'] = 'ThinkPHP@gmail.com';              //要修改的数据对象属性赋值
$User->where('id=5')->data($data)->save();          //根据条件保存修改的数据
```

第三种方法是针对某个字段的值，应用 setField()方法进行更新。例如，更新数据表中字段 name 的值，条件是 ID 为 5 的记录。

```
$User = M("User");                                  //实例化 User 对象
$User-> where('id=5')->setField('name','ThinkPHP'); //更改用户的 name 值
```

如果要更新多个字段的值，也可以应用 setField()方法，只需要传入数组即可，例如：

```
$User = M("User");                                  //实例化 User 对象
//更改用户的 name 和 email 的值
$User-> where('id=5')->setField(array('name','email'),array('ThinkPHP','ThinkPHP@gmail.com'));
```

第四种，应用 setInc()和 setDec()方法对统计字段（通常指的是数字类型）中的值进行增减操作。例如，对指定用户的积分进行增、减操作。

```
$User = M("User");                                  //实例化 User 对象
$User->setInc('score','id=5',3);                    //用户的积分加 3
$User->setInc('score','id=5');                      //用户的积分加 1
```

```
$User->setDec('score','id=5',5);                    //用户的积分减 5
$User->setDec('score','id=5');                      //用户的积分减 1
```

4．删除数据

在 ThinkPHP 中使用 delete()方法删除数据库中的记录，同样可以使用连贯操作进行删除操作。例如，删除数据表中 id 为 5 的记录。

```
$User = M("User");                                  //实例化 User 对象
$User->where('id=5')->delete();                     //删除 id 为 5 的用户数据
```

delete()方法可以用于删除单个或者多个数据，主要取决于删除条件，也就是 where()方法的参数，也可以用 order()和 limit()方法来限制要删除的个数。例如，删除所有状态为 0 的 5 个用户数据并按照创建时间排序。

```
$User = M("User");                                  //实例化 User 对象
$User->where('status=0')->order('create_time')->limit('5')->delete();
```

例 24.9 应用 ThinkPHP 中的 CURD 操作，实现对用户信息的查询、更新和删除操作。**（实例位置：光盘\TM\Instance\24\24.9）**

这里直接讲解在控制器中如何定义 index()方法完成循环输出数据库中的数据；定义 update()方法完成数据的更新；定义 delete()方法实现数据的删除。

其操作步骤如下。

（1）定位到 App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，以记录的 id 值为条件，降幂循环输出 10 条记录。其代码如下：

```
<?php
header("Content-Type:text/html; charset=utf-8");    //设置页面编码格式
class IndexAction extends Action{
	public function index(){
		$db = M('User');                            //实例化模型类，参数数据表名称不包含前缀
		$select = $db->order('id desc')->limit(10)->select();
		$this->assign('select',$select);            //模板变量赋值
		$this->display();                           //指定模板页
	}
}
```

定义 update()方法，首先根据超级链接传递 id 值执行查询，查询出指定的数据，并且将查询结果赋给指定的模板变量。然后，判断表单提交的 id 值是否存在，如果存在则以 id 为条件，对指定的数据进行更新操作。其关键代码如下：

```
	public function update(){
		$db = M('User');                            //实例化模型类，参数数据表名称不包含前缀
		$select = $db->where('id='.$_GET['id'])->select();
		$this->assign('select',$select);            //模板变量赋值
		$this->display(update);                     //指定模板页
		if(isset($_POST['id'])){
			$data['user'] = $_POST['user'];         //要修改的数据对象属性赋值
			$data['pass'] = md5($_POST['pass']);
			$data['address'] = $_POST['address'];
			$result=$db->where('id='.$_POST['id'])->save($data);     //根据条件保存修改的数据
			if($result){
				$this->redirect('Index/index','', 2,'数据更新成功');     //页面重定向
			}
		}

	}
```

定义 delete()方法，根据超链接传递的 id 值，删除数据库中指定的记录。其关键代码如下：

```
        public function delete(){
                $db = M('User');                                        //实例化模型类，参数数据表名称不包含前缀
                $result=$db->where('id='.$_GET['id'])->delete();        //删除 id 为 5 的用户数据
                if($result){
                        $this->redirect('Index/index','', 2,'数据删除成功');    //页面重定向
                }
        }
}
?>
```

（2）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，应用 ThinkPHP 内置模板引擎中的 foreach 标签循环输出模板变量传递的数据；创建更新和删除超链接，将指定记录的 id 作为参数进行传递。其关键代码如下：

```
<foreach name='select' item='user' >
   <tr class="content">
      <td bgcolor="#FFFFFF"> {$user.id}</td>
      <td bgcolor="#FFFFFF"> {$user.user}</td>
      <td bgcolor="#FFFFFF"> {$user.address}</td>
      <td bgcolor="#FFFFFF"><a href="__URL__/update?id={$user.id}">更新</a>/<a href="__URL__/delete?id=
{$user.id}">删除</a></td>
   </tr>
   </foreach>
```

（3）在 Index 模块文件夹下，编辑 update.html 模板文件，创建表单，将从模板变量中读取的数据作为表单元素的默认值进行输出，用表单中数据提交到控制器的 update()方法完成数据的更新操作。其关键代码如下：

```
<form id="form2" name="form2" method="post" action="__URL__/update">
<table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor="#FFFFFF">
<foreach name='select' item='user' >
   <tr class="content">
      <td bgcolor="#FFFFFF" class="right" width="103">名称：</td>
      <td bgcolor="#FFFFFF" width="289"> <input type="hidden" name="id" id="hiddenField" value="{$user.id}"
/><input name="user" type="text" id="user" size="20" value="{$user.user}" /></td>
      </tr>
   <tr class="content">
      <td bgcolor="#FFFFFF" class="right">密码：</td>
      <td bgcolor="#FFFFFF"><input name="pass" type="password" id="pass" size="20" value="{$user.pass}" />
       </td>
      </tr>
   <tr class="content">
      <td bgcolor="#FFFFFF" class="right"> 地址：</td>
      <td bgcolor="#FFFFFF"> 
         <input name="address" type="text" id="address" size="30" value="{$user.address}" />
      </td>
      </tr>
   <tr class="content">
      <td bgcolor="#FFFFFF"><input type="submit" name="button" id="button" value="更新" /></td>
   </tr>
   </foreach>
</table>
</form>
```

其运行效果如图 24.17 所示。

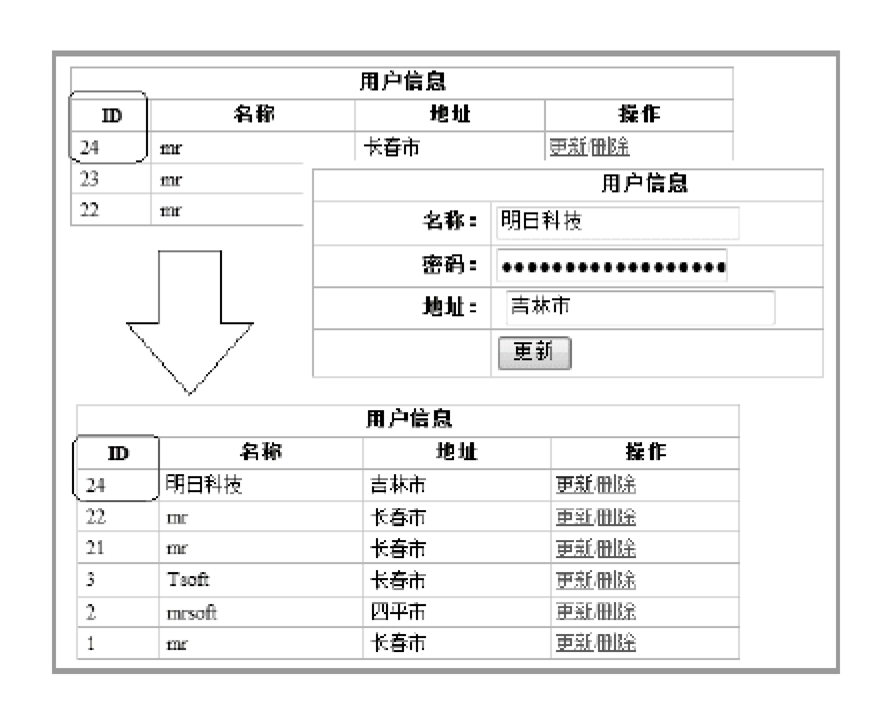


图 24.17　数据更新和删除

24.6　ThinkPHP 的视图

视频讲解：光盘\TM\Video\第 24 章\ThinkPHP 的视图.exe

在 ThinkPHP 里面，视图由两个部分组成：View 类和模板文件。Action 控制器直接与 View 视图类进行交互，把要输出的数据通过模板变量赋值的方式传递到视图类，而具体的输出工作则交由 View 视图类来进行，同时视图类还完成了一些辅助的工作，包括调用模板引擎、布局渲染、输出替换、页面 Trace 等功能。为了方便使用，在 Action 类中封装了 View 类的一些输出方法，如 display、fetch、assign、trace 和 buildHtml 等方法，这些方法的原型都在 View 视图类里面。

24.6.1　模板定义

为了对模板文件进行更加有效的管理，ThinkPHP 对模板文件进行了目录划分，默认的模板文件定义规则是：

```
模板目录/模板主题/[分组名/]模块名/操作名+模板后缀
```

模板目录默认是项目下面的 Tpl，模板主题默认是 default，模板主题功能是为了多模板切换而设计的，如果有多个模板主题，则可以使用 DEFAULT_THEME 参数设置默认的模板主题名。

在每个模板主题下面，是以项目的模块名为目录，然后是每个模块的具体操作模板文件，例如，User 模块的 Add 操作对应的模板文件是 Tpl/default/User/add.html。

模板文件的默认后缀是.html，后缀可以通过 TMPL_TEMPLATE_SUFFIX 来配置。

如果项目启用模块分组功能（假设 User 模块属于 Home 分组），那么默认对应的模板文件就会发生变化，变为：Tpl/default/Home/User/add.html。

分组功能可以通过 TMPL_FILE_DEPR 参数来配置，进而简化模板的目录层次。例如，设置 TMPL_FILE_DEPR 等于“_”，那么默认的模板文件就变成：Tpl/default/Home/User_add.html。

说明 正是因为系统有了这样一种模板文件自动识别的规则，display()方法才可以无需带任何参数就输出对应的模板。

24.6.2　模板赋值

模板赋值是在 Action 控制器中完成的，通过 assign()方法将控制器中获取的数据赋给模板变量。例如：



```
$this->assign('name',$value);
```

如果要同时输出多个模板变量，可以使用数组的方式进行赋值。

```
$array = array();
$array['name']      =     'thinkphp';
$array['email']     =     'liu21st@gmail.com';
$array['phone']      =     '12335678';
$this->assign($array);
```

这样，就可以在模板文件中同时输出 name、email 和 phone 3 个变量了。

24.6.3　指定模板文件

模板变量赋值后就需要调用模板文件来输出相关的变量，模板调用应用的是 display()方法。下面讲解如何通过 display()方法完成对模板的调用，如表 24.4 所示。

表 24.4　display()方法的应用

语 法 格 式	描　　述
display('操作名')	调用当前模块的其他操作模板。例如，当前是 User 模块下面的 read 操作，而要调用 User 模块的 edit 操作模板，$this->display('edit');
display('分组名:模块名:操作名')	调用其他模块的操作模板，其中分组名是可选的。例如，当前是 User 模块，要调用 Member 模块的 read 操作模板，使用：$this->display('Member:read'); 如果要调用分组 Admin 的 Member 模块的 read 操作模板，使用： $this->display('Admin:Member:read');
display('主题名@模块名:操作名')	调用其他主题的操作模板。例如，调用 Admin 主题的 User 模块的 edit 操作模板，使用：$this->display('Admin@User:edit');。此种方式需要指定模块和操作名
display('模板文件名')	直接全路径输出模板。例如，直接输出当前的 Public 目录下面的 menu.html 模板文件，使用： $this->display('./Public/menu.html');。这种方式需要指定模板路径和后缀，这里的 Public 目录是位于当前项目入口文件位置下面。如果是其他的后缀文件，也支持直接输出，例如，$this->display('./Public/menu.tpl'); 只要./Public/menu.tpl 是一个实际存在的模板文件。如果使用的是相对路径的话，要注意当前位置是相对于项目的入口文件，而不是模板目录
display('模板文件名', 'charset')	设置模板页的编码。例如，设置指定模板页的编码为 gbk，$this->display('Member:read', 'gbk');
display('模板文件名', 'charset', 'format')	设置指定模板文件的编码和格式。例如，设置模板文件为 utf-8 编码，设置文件为 XML 格式。其应用如下：$this->display('Member:read', 'utf-8', 'text/xml');

说明 在第二种用法中，不需要写模板文件的路径和后缀，严格来说，这里面的模块名和操作名并不一定需要有对应的模块或者操作，只是一个目录名称和文件名称而已。例如，项目中可能没有 Public 模块，更没有 Public 模块的 menu 操作，但是一样可以使用“$this->display('Public:menu');”语句输出这个模板文件。

模板变量赋值后，在指定的模板文件中进行输出，具体的输出方法需要根据选择的模板引擎来决定。如果使用的是内置的模板引擎，请参考 ThinkPHP 开发完全手册模板指南中的内容；如果使用 PHP 本身作为模板引擎，则直接在模板文件里面输出。例如，<?php echo $name.'['.$email.' '.$phone.']';?>。

24.6.4 特殊字符串替换

在进行模板输出之前，系统还会对模板的特殊字符串进行替换，实现模板输出的替换和过滤。这个机制可以使得模板文件的定义更加方便，默认的替换规则如表 24.5 所示。

表 24.5 模板中特殊字符串的替换规则

特殊字符串	替 换 描 述
../Public	被替换成当前项目的公共模板目录。通常是：/项目目录/Tpl/default/Public/
__PUBLIC__	被替换成当前网站的公共目录 通常是：/Public/
__TMPL__	替换成项目的模板目录。通常是：/项目目录/Tpl/default/
__ROOT__	会替换成当前网站的地址（不含域名）
__APP__	替换成当前项目的 URL 地址（不含域名）
__URL__	替换成当前模块的 URL 地址（不含域名）
__ACTION__	替换成当前操作的 URL 地址（不含域名）
__SELF__	替换成当前的页面 URL

说明 这些特殊的字符串是严格区别大小写的，并且这些特殊字符串的替换规则是可以更改或者增加的。只要在项目配置文件中配置 TMPL_PARSE_STRING 就可以完成。如果有相同的数组索引，就会更改系统的默认规则。例如：

```
TMPL_PARSE_STRING   => array(
  '__PUBLIC__' => '/Common',                         //更改默认的__PUBLIC__替换规则
  '__UPLOAD__' => '/Public/Uploads/',                //增加新的上传路径替换规则
)
```

例 24.10 实现用户登录功能，将登录用户的信息存储到 SESSION 变量中，应用 ThinkPHP 中提供的分页扩展类和 Page 方法完成数据的分页输出。（**实例位置：光盘\TM\Instance\24\24.10**）

其操作步骤如下。

（1）定位到 App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 session 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();                                                    //初始化 session 变量
header("Content-Type:text/html; charset=utf-8");                    //设置页面编码格式
class IndexAction extends Action{
	public function index(){
		if(isset($_POST['user'])){
			if(isset($_POST['user']) && isset($_POST['pass'])){
				$db = M();                                              //实例化模型类，参数数据表名称不包含前缀
				$select = $db->query("select * from think_user where user='".$_POST['user']."' and
pass='".$_POST['pass']."'");                                        //执行查询语句，验证用户名和密码是否正确
				if($select){
					$_SESSION['admin']=$_POST['user'];              //将登录用户名存储到 session 中
					$this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！');  //页面重定向
				}else{
```

```
                    $this->redirect('Index/index','', 2,'用户名或者密码不正确！');          //页面重定向
                }
            }else{
                $this->redirect('Index/index','', 2,'用户名、密码不能为空！');             //页面重定向
            }
        }
        $this->display();
    }
}
?>
```

定义 main()方法，载入分页类，完成数据库中数据的分页查询，并且将查询结果赋给模板变量。其代码如下：

```
public function main(){
        $db = M('User');                                   //实例化模型类，参数数据表名称不包含前缀
        //进行分页数据查询，注意，Page()方法的参数的前面部分是当前的页数，使用 $_GET[p]获取
        if(isset($_GET['p'])){                             //判断分页变量是否存在
            $p=$_GET['p'];
        }else{
            $p=1;
        }
        $list = $db->where('address='."'长春市'")->order('id desc')->page($p.',1')->select();   //查询数据
        $this->assign('select',$list);                                                        //赋值数据集
        import("ORG.Util.Page");                                    //导入分页类
        $count = $db->where('address='."'长春市'")->count();       //查询满足要求的总记录数
        $Page = new Page($count,1);                        //实例化分页类，传入总记录数和每页显示的记录数
        $show = $Page->show();                             //分页显示输出
        $this->assign('page',$show);                       //赋值分页输出
        $this->display(main);                              //输出模板
    }
```

定义 validatorcode()方法，应用 GD 库中的函数，根据超链接传递的值生成用户登录的验证码。其代码如下：

```
    public function validatorcode(){
        header('content-type:image/png');                                       //定义标题 PNG 格式图像
        $im = imagecreate(65, 25);                                              //定义画布
        imagefill($im, 0, 0, imagecolorallocate($im, 200, 200, 200));           //区域填充
        $validatorCode = $_GET['code'];                                         //获取提交的值
        imagestring($im, rand(3, 5), 10, 3, substr($validatorCode, 0, 1), imagecolorallocate($im, 0, rand(0,
255), rand(0, 255)));
        imagestring($im, rand(3, 5), 25, 6, substr($validatorCode, 1, 1), imagecolorallocate($im, rand(0, 255),
0, rand(0, 255)));
        imagestring($im, rand(3, 5), 36, 9, substr($validatorCode, 2, 1), imagecolorallocate($im, rand(0, 255),
rand(0, 255), 0));
        imagestring($im, rand(3, 5), 48, 12, substr($validatorCode, 3, 1), imagecolorallocate($im, 0, rand(0,
255), rand(0, 255)));
        imagepng($im);                                                          //生成 PNG 图像
        imagedestroy();                                                         //销毁图像
    }
```

（2）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文件和 JavaScript 文件，创建表单，完成用户登录信息的提交操作。其关键代码如下：

```
<link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
<js href="__ROOT__/Public/Js/check.js" />
```

```
<form name="form1" method="post"   action="__URL__/index" onSubmit="return chkinput(this)" >
<table width="265" border="0" cellspacing="0" cellpadding="0">
     <tr>
          <td class="title" id="td">用户名：</td>
          <td><input name="user" type="text" size="15" /></td>
     </tr>
     <tr>
          <td class="title" id="td">密码：</td>
          <td><input name="pass" type="password" size="15" /></td>
     </tr>
     <tr>
          <td class="title" id="td">验证码：</td>
          <td>
          <input type="text" name="validatorCode" size="10" />
               <input type="hidden" name="defValidatorCode" value="" />
<script language="javascript">
     var num1=Math.round(Math.random()*10000000);                    //生成随机数
    var num=num1.toString().substr(0,4);                              //截取随机数的前 4 个字符
document.write("<img name=codeimg src='__URL__/validatorcode?code="+num+"'>");
                                                                       //将截取值传递到图像处理页中
    form1.defValidatorCode.value=num;                                 //将截取值赋给表单中的隐藏域
     function reCode(){                                               //定义方法,重新生成验证码
          var num1=Math.round(Math.random()*10000000);                //生成随机数
         var num=num1.toString().substr(0,4);                         //截取随机数
         document.codeimg.src="__URL__/validatorcode?code="+num;  //将截取值传递到图像处理页中
         form1.defValidatorCode.value=num;                            //将截取值赋给表单中的隐藏域
   }
</script>
               <a href="javascript:reCode()" class="content">看不清</a>
          </td>
     </tr>
</table>
     <input type="image" name="imageField" id="imageField" src="__ROOT__/Public/images/66_05.gif" />
</form>
```

（3）在 Index 模块文件夹下，编辑 main.html 文件，通过模板引擎中的 session 标签输出当前登录的用户名，通过 foreach 标签循环输出模板变量传递的数据，最后输出模板变量传递的分页超链接。其关键代码如下：

```
<table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor="#FFFFFF">
  <tr>
    <td colspan="3" bgcolor="#FFFFFF" class="title" align="center">当前登录用户：{$Think.session.admin}</td>
  </tr>
  <foreach name='select' item='user' >
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$user.id}</td>
    <td bgcolor="#FFFFFF"> {$user.user}</td>
    <td bgcolor="#FFFFFF"> {$user.address}</td>
  </tr>
  </foreach>
   <tr class="content">
    <td colspan="3" bgcolor="#FFFFFF"> {$page}</td>
    </tr>
</table>
```



其运行效果如图 24.18 所示。

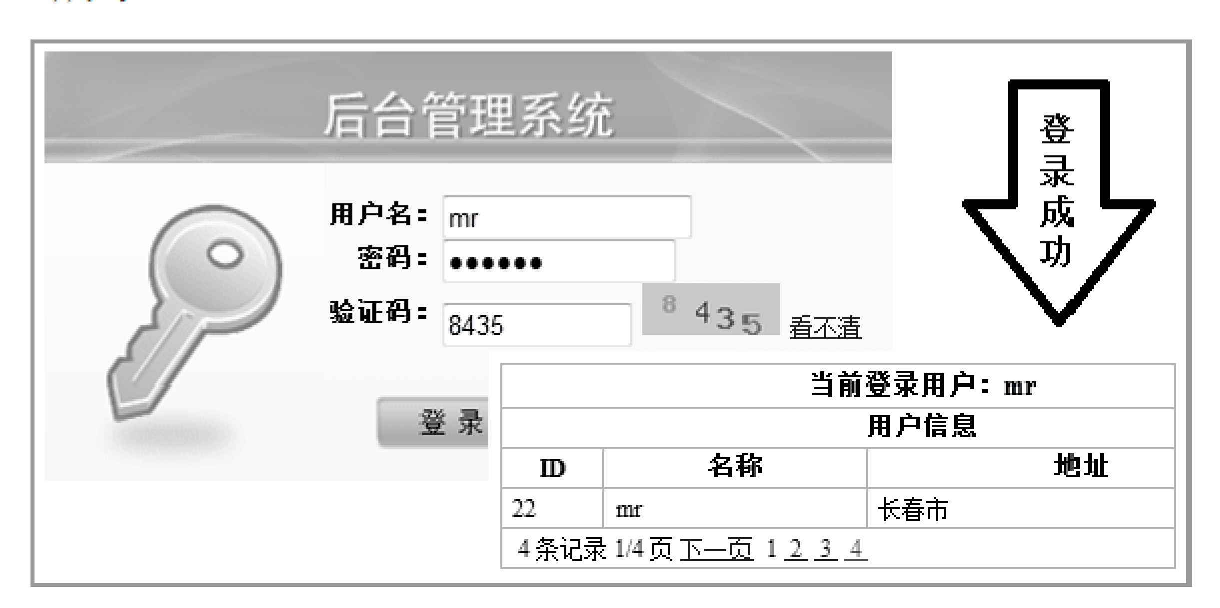


图 24.18　用户登录和数据的分页输出

24.7　内置 ThinkTemplate 模板引擎

视频讲解：光盘\TM\Video\第 24 章\内置 ThinkTemplate 模板引擎.exe

ThinkPHP 内置了一个基于 XML 的性能卓越的模板引擎 ThinkTemplate，这是一个专门为 ThinkPHP 服务的内置模板引擎。ThinkTemplate 是一个使用 XML 标签库技术的模板引擎，支持两种类型的模板标签（普通标签和 XML 标签），使用了动态编译和缓存技术，而且支持自定义标签库。

ThinkTemplate 模板引擎生成的编译文件默认存储于 Runtime/Cache 目录下面，以模板文件的 MD5 编码作为缓存文件名保存。

下面介绍一些 ThinkTemplate 模板引擎中的常用标签，如表 24.6 所示。

表 24.6　ThinkTemplate 模板引擎中的常用标签

标 签 名 称	应 用 描 述
{$name}	输出模板引擎中的变量。注意模板标签的“{”和“$”之间不能有任何的空格，否则标签无效

//输出$_SERVER 变量
{$Think.server.script_name }
//输出$_SESSION 变量
{$Think.session.session_id|md5 }
//输出$_GET 变量
{$Think.get.pageNumber }
//输出$_COOKIE 变量
{$Think.cookie.name }
{$Think.now } //现在时间
{$Think.template|basename } //模板页面
	系统变量。除了常规变量的输出外，模板引擎还支持系统变量和系统常量，以及系统特殊变量的输出。它们的输出不需要事先赋值给某个模板变量。系统变量的输出必须以$Think. 打头，并且仍然支持使用函数

{$变量|default="默认值"}	默认值输出。如果输出的模板变量没有值，但是需要在显示时赋予一个默认值的话，可以使用 default 语法。例如，{$user.nickname\|default="明日科技"}对系统变量的输出也可以支持默认值，例如，{$Think.post.name|default="名称为空"}

续表

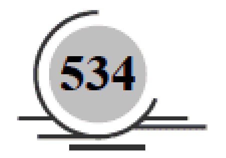
标 签 名 称	应 用 描 述

//使用完整文件名包含
<include file="完整模板文件名" />
//包含当前模块的其他操作模板文件
<include file="操作名" />
//包含其他模块的操作模板
<include file="模块名:操作名" />
//包含其他模板主题的模块操作模板
<include file="主题名@模块名:操作名" />
//用变量控制要导入的模板
<include file="$变量名" />
	使用 Include 标签来包含外部的模板文件
	完整文件名的包含，例如，<include file="./Tpl/default/Public/header.html" />。这种情况下，模板文件名必须包含后缀。使用完整文件名包含时，特别要注意文件包含指的是服务器端包含，而不是包含一个 URL 地址，也就是说 file 参数的写法是服务器端的路径，如果使用相对路径的话，是基于项目的入口文件位置。
	用变量包含。例如，<include file="$tplName" />。给$tplName 赋不同的值就可以包含不同的模板文件，变量的值的用法和上面的用法相同。
	注意：由于模板解析的特点，从入口模板开始解析，如果外部模板有所更改，模板引擎并不会重新编译模板，除非缓存已经过期。如果修改了包含的外部模板文件后，需要把模块的缓存目录清空，否则无法生效

<import type='js' file="Js.Util.Array" /> <import type='css' file="Css.common" /> <load href="../Public/Js/Common.js" /> <load href="../Public/Css/common.css" /> <js href="__PUBLIC__/Js/Common.js" /> <css href="../Public/Css/common.css" />	导入文件。系统提供专门的是 import 标签和 load 标签完成文件的导入操作。第一个是 import 标签 ，导入方式采用类似 ThinkPHP 的 import 函数的命名空间方式。import 标签默认的起始路径是网站的 Public 目录，如果需要指定其他的目录，可以使用 basepath 属性，例如，<import file="Js.Util.Array" basepath="./Common" /> 第二个是 load 标签，通过文件方式导入当前项目的公共 JS 或者 CSS。在 href 属性中可以使用特殊模板标签替换，例如，<load href="__PUBLIC__/Js/Common.js" /> Load 标签可以无需指定 type 属性，系统会根据后缀自动判断。 系统还提供了两个标签别名 js 和 css 用法和 load 一致

<volist name="list" id="vo" offset="5" length='10'>
{$vo.name}
</volist>
	volist 标签主要用于在模板中循环输出数据集或者多维数组。标签参数如下。
	name：表示模板赋值的变量名称，因此不可随意在模板文件中改变。
	id：表示当前的循环变量，可以随意指定，但确保不要和 name 属性冲突。
	支持输出部分数据，例如输出其中的第 5～15 条记录。
	offset：表示记录的起始位置。
	length：表示记录的长度。
	mod：控制输出记录的奇偶性，还可以控制在指定的记录换行。例如：
	//输出偶数记录
	<volist name="list" id="vo" mod="2" >
	<eq name="mod" value="1">{$vo.name}</eq>
	</volist>
	//控制一定记录的换行
	<volist name="list" id="vo" mod="5" >
	{$vo.name}
	<eq name="mod" value="4">
	</eq>
	</volist>

<foreach name="list" item="vo" > {$vo.id} {$vo.name} </foreach>	foreach 标签用于循环输出，它比 volist 标签简洁，没有 volist 标签那么多的功能。其优势是可以对对象进行遍历输出，而 volist 标签通常是用于输出数组



续表

标 签 名 称	应 用 描 述
<switch name="变量" > <case value="值 1">输出内容 1</case> <case value="值 2">输出内容 2</case> <default/>默认情况 </switch>	switch 标签，类似于 PHP 中的 switch 语句。 其中 name 属性可以使用函数以及系统变量，例如： <switch name="Think.get.userId|abs"> <case value="1">admin</case> <default />default </switch> 对于 case 的 value 属性可以支持多个条件的判断，使用“|”进行分割，例如： <switch name="Think.get.type"> <case value="gif|png|jpg">图像格式</case> <default />其他格式 </switch> 表示如果$_GET["type"] 是 gif、png 或者 jpg 的话，就判断为图像格式。 也可以对 case 的 value 属性使用变量，例如： <switch name="User.userId"> <case value="$adminId">admin</case> <case value="$memberId">member</case> <default />default </switch> 使用变量方式的情况下，不再支持多个条件的同时判断

例 24.11　仍然以用户登录和数据的输出为背景，应用 ThinkPHP 中提供的验证码类和分页类生成验证码，完成数据的分页输出。**（实例位置：光盘\TM\Instance\24\24.11）**

其操作步骤如下。

（1）定位到 App\Lib\Action\目录下，编写项目控制器。创建 Index 模块，继承系统的 Action 基础类，定义 index()方法，验证 session 变量存储的验证码与用户提交的验证码是否相同，验证用户提交的用户名和密码是否正确，如果正确则将登录用户名存储到 session 变量中，并且将网页重定向到 main.html 页面。其代码如下：

```
<?php
session_start();
header("Content-Type:text/html; charset=utf-8");                              //设置页面编码格式
class IndexAction extends Action{
	public function index(){
		if(isset($_POST['user'])){
			if(isset($_POST['user']) && isset($_POST['pass']) && isset($_POST['validatorCode'])){
				if($_SESSION['verify'] == md5($_POST['validatorCode'])) {	//验证验证码是否正确
					$db = M();			//实例化模型类，参数数据表名称不包含前缀
					$select = $db->query("select * from think_user where user='".$_POST['user']."' and
pass='".$_POST['pass']."'");	//执行查询语句，验证用户名和密码是否正确
					if($select){
						$_SESSION['admin']=$_POST['user'];
						$this->redirect('Index/main','', 2,'用户 '.$_POST['user'].' 登录成功！'); //页面重定向
					}else{
						$this->redirect('Index/index','', 2,'用户名或者密码不正确！');		//页面重定向
					}
				}else{
					$this->redirect('Index/index','', 2,'验证码不正确！');			//页面重定向
```

```
            }
        }else{
            $this->redirect('Index/index','', 2,'用户名、密码不能为空！');            //页面重定向
        }
    }
    $this->display();
}
```

定义 main()方法，载入分页类，完成数据库中数据的分页查询，并且将查询结果赋给模板变量。这里应用的是 Page 类和 limit 方法完成数据的分页输出。其代码如下：

```
public function main(){
        $db = M('User');                                    //实例化模型类，参数数据表名称不包含前缀
        import("ORG.Util.Page");                            //导入分页类
        $count = $db->count();                              //统计总记录数
        //$count = $User->where("status=1")->count();       //查询满足要求的总记录数
        $Page = new Page($count,1);                         //实例化分页类，传入总记录数和每页显示的记录数
        $show = $Page->show();                              //分页显示输出
        //进行分页数据查询，注意，limit()方法的参数要使用 Page 类的属性
        $list = $db->order('id')->limit($Page->firstRow.','.$Page->listRows)->select();
        $this->assign('select',$list);                      //赋值数据集
        $this->assign('page',$show);                        //赋值分页输出
        $this->display(main);                               //输出模板
}
```

定义 verify()方法，载入 ThinkPHP 中提供的验证码扩展类，调用 buildImageVerify()方法生成验证码。其代码如下：

```
public function verify(){
    import("ORG.Util.Image");                               //载入验证码类
    image::buildImageVerify(4,5,'png','65','30','verify');  //生成验证码
}
```

说明 buildImageVerify ()方法的语法如下：

```
buildImageVerify($length,$mode,$type,$width,$height,$verifyName)
```

参数说明如表 24.7 所示。

表 24.7 验证码类中 buildImageVerify()方法说明

参 数	说 明
length	验证码的长度，默认为 4 位数
mode	验证字符串的类型，默认为数字，其他支持类型有 0 字母 1 数字 2 大写字母 3 小写字母 4 中文 5 混合（去掉了容易混淆的字符 oOLl 和数字 01）
type	验证码的图片类型，默认为 png
width	验证码的宽度，默认会自动根据验证码长度自动计算
height	验证码的高度，默认为 22
verifyName	验证码的 SESSION 记录名称，默认为 verify

生成验证码之后，需要在模板页中通过输出生成的验证码图像。在控制器中通过如下代码：

```
if($_SESSION['verify'] != md5($_POST['verify'])) {
    $this->error('验证码错误！');
}
```

（2）定位到 App\Tpl\default 目录下，创建 Index 模块文件夹。编辑 index 操作的模板文件 index.html，载入 CSS 样式文件和 JavaScript 文件，创建表单，完成用户登录信息的提交操作，通过 img 标签输出生成的验证码。其关键代码如下：

```
<link href="__ROOT__/Public/Css/style.css" rel="stylesheet" type="text/css" />
<js href="__ROOT__/Public/Js/check.js" />
<form name="form1" method="post"   action="__URL__/index" onSubmit="return chkinput(this)" >
<table width="265" border="0" cellspacing="0" cellpadding="0">
      <tr>
            <td class="title" id="td">用户名：</td>
            <td><input name="user" type="text" size="15" /></td>
      </tr>
      <tr>
            <td class="title" id="td">密码：</td>
            <td><input name="pass" type="password" size="15" /></td>
      </tr>
      <tr>
            <td class="title" id="td">验证码：</td>
            <td><input type="text" name="validatorCode" size="10" /></td>
            <td><img src="__APP__/Index/verify/" /></td>
      </tr>
</table>
</form>
```

（3）在 Index 模块文件夹下，编辑 main.html 文件，通过模板引擎中的 session 标签输出当前登录的用户名，通过 foreach 标签循环输出模板变量传递的数据，最后输出模板变量传递的分页超链接。其关键代码如下：

```
<table width="405" border="1" cellpadding="1" cellspacing="1" bgcolor="#99CC33" bordercolor="#FFFFFF">
  <tr>
    <td colspan="3" bgcolor="#FFFFFF" class="title" align="center">当前登录用户：{$Think.session.admin}</td>
  </tr>
  <foreach name='select' item='user' >
  <tr class="content">
    <td bgcolor="#FFFFFF"> {$user.id}</td>
    <td bgcolor="#FFFFFF"> {$user.user}</td>
    <td bgcolor="#FFFFFF"> {$user.address}</td>
  </tr>
  </foreach>
   <tr class="content">
    <td colspan="3" bgcolor="#FFFFFF"> {$page}</td>
    </tr>
</table>
```

说明 这里应用的验证码是区分字母的大小写的。

其运行效果如图 24.19 所示。

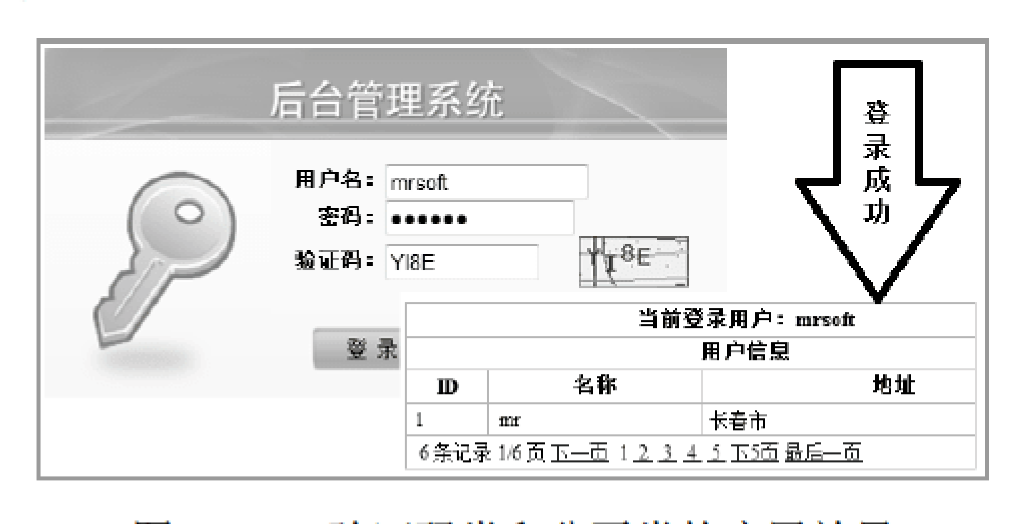


图 24.19　验证码类和分页类的应用效果

24.8　小　　结

本章主要介绍 ThinkPHP 框架的下载、架构、配置、控制器、模型以及视图。并通过实例，对 ThinkPHP 的各种应用进行了讲解，以此来增加读者对 ThinkPHP 的理解。希望通过本章的学习，读者能够掌握 ThinkPHP 技术，将其灵活运用到实际的网站开发中。

24.9　学习成果检验

1．通过 ThinkPHP 中的扩展类生成中文验证码，其关键是载入 ThinkPHP\Lib\ORG\Util\Image.class.php 中的 Image 类，调用其中的 GBVerify()方法，生成中文验证码。其运行效果如图 24.20 所示。**（实例位置：光盘\TM\Instance\24\24.12）**

注意 在 ThinkPHP 2.0 版本提供的 Image 类中，在定义 GBVerify()方法时有一处错误，需要读者手动修改。其中“$codex = msubstr($code, $i, 1);”对字符串进行截取调用的 msubstr()方法不存在。正确的调用方法是“$codex = String::msubstr($code, $i, 1);”。

2．应用 ThinkPHP 中的文件上传扩展类实现文件上传的功能，上传 JPG、GIF 或者 PNG 格式的图片到服务器指定的文件夹下，其运行效果如图 24.21 所示。**（实例位置：光盘\TM\Instance\24\24.13）**



图 24.20　中文验证码



图 24.21　应用 ThinkPHP 中的扩展类上传文件

第25章

Zend Framework 框架

（视频讲解：39 分钟）

Zend Framework（简称 ZF）是由 Zend 公司支持开发的完全基于 PHP 5 的开源框架，采用 MVC 架构模式来分离应用程序中不同的部分，便于程序的开发和维护。其拥有丰富的组件支持，模块化的结构设计，易于扩展和完善的文档资料以及灵活的架构设计。

通过阅读本章内容，你可以：

- 了解 Zend Framework 的 MVC
- 熟悉 Zend Framework 的 MVC 环境搭建
- 掌握 Zend_Auth 身份认证
- 掌握 Zend_Db 数据库操作
- 掌握 Zend_File 文件控制
- 掌握 Zend_Layout 网站布局
- 掌握 Zend_Paginator 分页

25.1　Zend Framework 的 MVC 介绍

25.1.1　Zend Framework 概述

Zend Framework 框架由 Zend 公司主创，同时 Google、Microsoft 和 StrikeIron 作为合作伙伴为 Zend Framework 提供了大量技术和 Web 服务接口。

Zend Framework 中的组件都是独立的，都不依赖于其他组件，这样的松藕合结构可以让开发者独立使用组件。

Zend Framework 版本更新速度非常快，所包含的组件也不断增加，所以说 Zend Framework 框架可以让一个从事 PHP 工作的人始终保持与最新技术的接触和学习。

Zend Framework 有自己成熟的社区，关于 Zend Framework 的任何问题都能在社区里找到答案。

25.1.2　Zend Framework 常用组件

由于 Zend Framework 的组件都是非常独立的（这种情况称为松耦合），所以在开发 Web 程序或者软件时可以独立使用这些组件的组合。表 25.1 列举了 Zend Framework 的常用组件及功能。

表 25.1　Zend Framework 常用组件及功能

组 件 名 称	功　　能
Zend_Acl	权限控制
Zend_Auth	主要用于认证
Zend_Cache	为应用程序提供缓存支持
Zend_Config	应用程序的配置数据参数
Zend_Db	提供 Zend Framework 与 MySQL 的数据库操作方式
Zend_File	开发者控制文件的上传和下载
Zend_Layout	实现应用程序的视图布局
Zend_Mail	实现 Zend Framework 发送和接收 Email
Zend_Paginator	提供数据的分页显示
Zend_Registry	提供对象注册表，相当于全局变量在程序中的使用
Zend_Session	提供 Zend Framework 的 session 控制
Zend_Validate	提供一组通用校验器

25.1.3　MVC 原理

MVC 是一种经典的程序设计理念，它将应用程序分为 3 部分：模型层（Model）、视图层（View）和控制层（Controller），MVC 即是这 3 部分英文名称字母的缩写。

MVC 设计模式产生的原因如下：

应用程序中用来完成任务的代码—— 模型层（也叫业务逻辑），通常是程序中相对稳定的部分，重用率高。而与用户交互界面—— 视图层，却经常改变。如果因需求变动而不得不对业务逻辑代码进行修改，或者要在不同的模块中应用相同的功能而要重复编写业务逻辑代码，不仅降低整体程序的开发进度，也会



使未来的维护变得非常困难。因此，将业务逻辑代码与外观分离，能够更方便地根据需求改进程序，这就是 MVC 设计模式。

在 PHP Web 开发中，MVC 设计模式的各自功能及相互关系如图 25.1 所示。

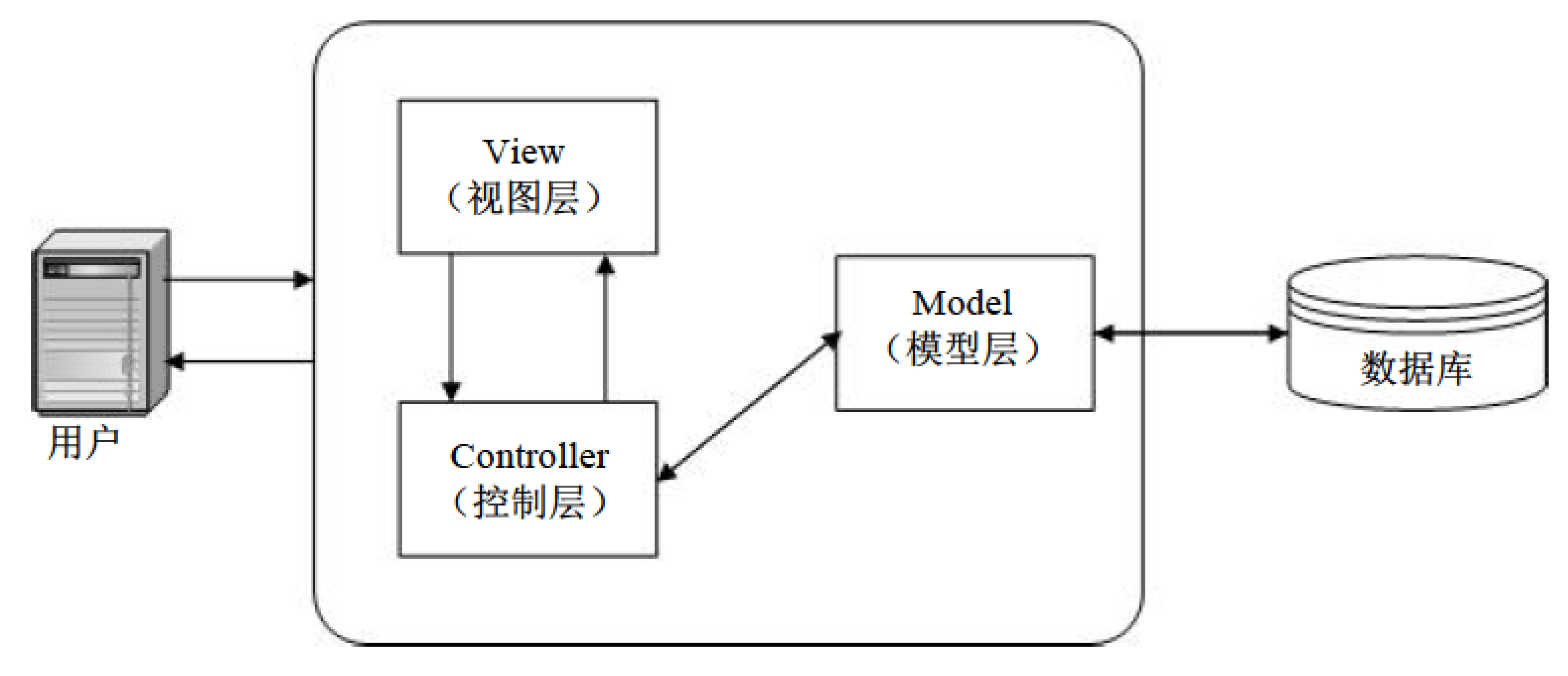


图 25.1　MVC 关系图

☑　模型层（Model）

模型层是应用程序的核心部分，它可以是一个实体对象或一种业务逻辑，之所以称为模型，是因为它在应用程序中有更好的重用性和扩展性。

☑　视图层（View）

视图层提供应用程序与用户之间的交互界面，在 MVC 理论中，这一层并不包含任何业务逻辑，仅提供一种与用户交互的视图。

☑　控制层（Controller）

控制层用于对程序中的请求进行控制，其作用就像国家的宏观调控，它可以选择调用哪些视图或者模型。

25.2　Zend Framework 的 MVC 环境搭建

视频讲解：光盘\TM\Video\第 25 章\Zend Framework 的 MVC 环境搭建.exe

利用 Zend Framework 开发项目，首先必须保证 PHP 运行环境对 Zend Framework 框架的支持。Zend Framework 环境的搭建主要根据 PHP 环境而来，所以必须搭建 Apache 和 MySQL。

25.2.1　环境配置

搭建 Apache 和 MySQL，搭建的 Apache 中必须有 PDO_MYSQL 模块，如果安装环境中没有此模块，可以去 PHP 官方网站中下载。

1．配置 HTTPD.CONF

（1）进入 Apache 的 conf 目录下，使用编辑工具打开 httpd.conf 文件，定位到如下内容：

```
#LoadModule rewrite_module modules/mod_rewrite.so
```

（2）该句中的“#”符号用于在.conf 文件中注释掉该行。去掉“#”符号后，表示加载 mod_rewrite 模块。

（3）在 Apache 加载 mod_rewrite.so 模块后需要指定生效的目录，在 httpd.conf 文件中找到所有为“AllowOverride None”的字符串，将其修改为“AllowOverride All”，然后保存 httpd.conf 文件即可开启 mod_rewrite 功能。

技巧 在去掉 mod_rewrite.so 前面的“#”后，需要从 hhtpd.conf 文件开始位置重新查找，才能保证从全文中查找“AllowOverride None”。

2．配置 PHP.INI

Zend Framework 操作 MySQL 使用 PHP 自带的 PDO_MYSQL 模块。默认的 PHP 是不开启 PDO_MYSQL 模块的，所以必须重新对 PHP 环境进行配置。这里以 Windows 系统为例，在 C:\WINDOWS 下找到 php.ini 文件，定位到如下位置：

```
;extension=php_pdo.dll
......
;extension=php_pdo_mysql.dll
```

将这两行前面的“;”（ini 文件的注释符）去掉，并保存 php.ini 文件。然后定位到 PHP 安装目录的 ext 文件夹下，在该文件夹中查看是否存在 php_pdo_mysql.dll 和 php_pdo.dll 文件，如果不存在，则需下载并存入 ext 文件夹中。至此 PDO_MYSQL 扩展加载成功。

至此，重新启动 Apache 服务器，Zend Framework 运行环境配置成功。

注意 由于 Zend Framework 完全采用 PHP5 的面向对象编程方式实现，所以 PHP 版本必须为 PHP 5 以上。同时，为了发挥 Zend Framework 各个组件的功能，建议使用 PHP 5.2.6 及以上版本。如果是使用集成化安装包 AppServ 来搭建 PHP 开发环境，那么需要下载 AppServ 2.5.10 或者更高版本。

25.2.2 框架结构

要在项目开发中应用 Zend Framework 框架，只完成环境的配置是不够的，还要搭建一个完整的框架结构，才能在项目开发中发挥框架的作用。

自 Zend Framework 1.8 之后推出了 Zend_Application 组件，使搭建结构更加方便和系统化。如图 25.2 所示为一个相对完整的 Zend Framework 框架结构。

25.2.3 创建流程

了解框架的结构后，下面讲解使用 Zend Framework 建立多模块 MVC 框架结构的具体流程。Zend Framework 程序开发的基本流程如图 25.3 所示。

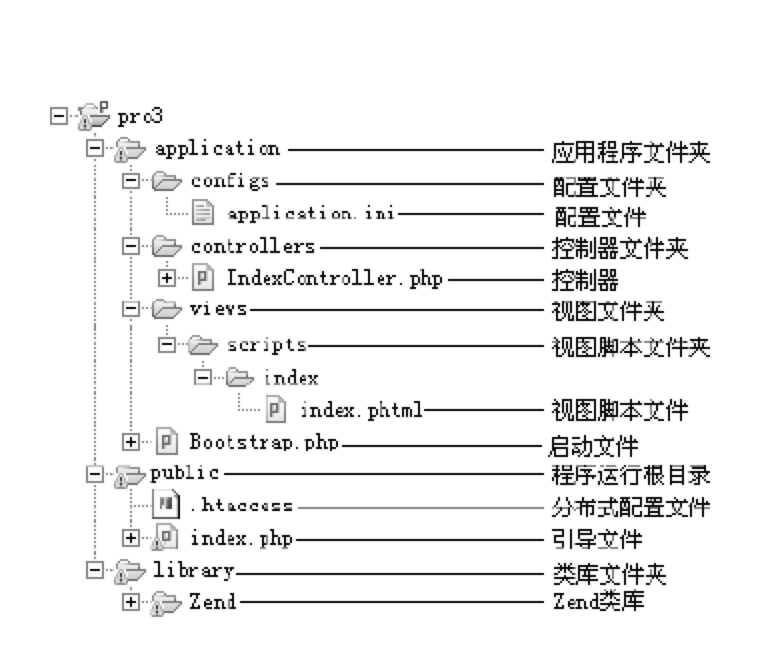


图 25.2　Zend Framework 框架结构

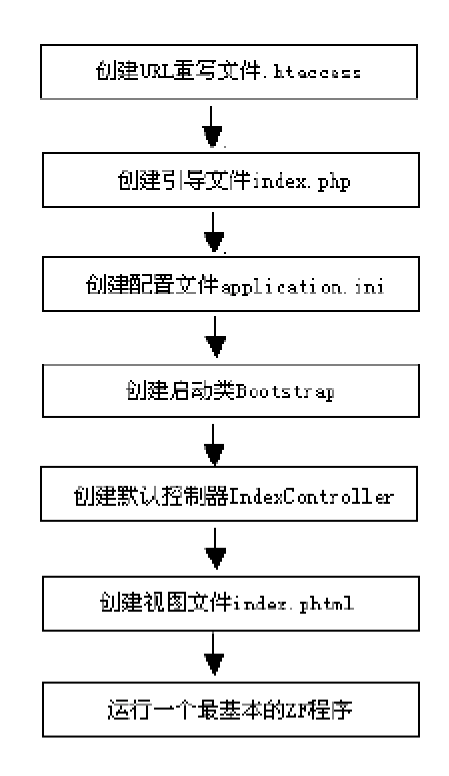


图 25.3　最基本的 Zend Framework 程序开发步骤



例 25.1　开发一个简单的实例，输出“Hello，Zend Framework”，熟悉 Zend Framework 框架的基本流程。（**实例位置：光盘\TM\Instance\25\25.1**）

（1）在 public 目录下建立 URL 重写文件.htaccess，并在该文件中输入 URL 重写规则，如下所示：

```
#开启 URL 重写
RewriteEngine on
#除扩展名为.js、.css、.gif、.jpg、.png、.bmp 的文件外，访问其他文件都转向到 index.php 引导文件
RewriteRule !\.(js|css|gif|jpg|png|bmp)$ index.php
```

技巧 由于.htaccess 文件没有文件主名，所以在 Windows 系统下无法直接命名，下面介绍两种创建.htaccess 文件的方法。

方法一：通过 Windows 系统的 copy con 命令创建，如图 25.4 所示。创建完成后按 Ctrl+Z 组合键退出编辑模式。



图 25.4　通过命令提示符创建.htaccess 文件

方法二：通过“记事本”文本编辑工具将文件另存为文件名为.htaccess 的文件，这里应该注意，保存文件时需要选择文件类型为所有文件。

（2）在 public 目录下创建 index.php 引导文件。代码如下：

```
defined('APPLICATION_PATH') || define('APPLICATION_PATH', realpath(dirname(__FILE__) . '/../application'));
                                                                    //应用路径
defined('APPLICATION_ENV') || define('APPLICATION_ENV', getenv('APPLICATION_ENV') ? getenv
('APPLICATION_ENV') : 'project');                                   //应用环境
$arrayIncludePath = array('.', realpath(dirname(__FILE__) . '/../../../library'));   //指定工程包含目录
set_include_path(implode(PATH_SEPARATOR, $arrayIncludePath));       //将指定路径包含到工程中
require_once 'Zend/Application.php';                                //包含 Application.php 文件
$application = new Zend_Application(APPLICATION_ENV, APPLICATION_PATH . '/configs/application.ini');
//实例化 Zend_Application 类
$application->bootstrap()->run();                                   //调用启动文件并运行项目
```

上述代码中，首先定义两个常量 APPLICATION_PATH 和 APPLICATION_ENV，分别用来保存应用路径和应用环境名，这样在工程的其他模块中就可以直接使用，然后将当前目录地址“.”和 Zend Framework 类库地址作为数组元素保存在名为$arrayIncludePath 的数组中，并使用 implode()函数将上述路径用 PATH_SEPARATOR 常量连接，最后使用 set_include_path()方法将路径导入到工程中，这样工程就可以直接找到这些路径下的文件。

Zend Framework 类库被导入到工程后，就可以使用 require_once 等文件包含语句包含 Zend 目录下的 Application.php 文件，之后实例化 Zend_Application 类，并通过引用该类的 bootstrap()方法调用启动类，最终通过调用 run()方法运行工程。

注意 在文件的结尾部分并没有加入“?>”，因为在文件的结尾它并不是必需的。这样就可以避免产生一些难于调控的错误问题。例如，在使用 header()函数来重新定向（redirect）时，若在其前面某个包含文件中“?>”的后面加上空格就会出现错误。

（3）在第（2）步实例化 Zend_Application 类时，为类的构造函数传入一个名为 application.ini 的文件，该文件是工程的配置文件。下面介绍 application.ini 文件的创建过程。其基本内容如下：

```
[project]
#设置错误级别
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1
#设置时区
phpSettings.date.timezone = 'Asia/Shanghai'
#配置启动类
bootstrap.path = APPLICATION_PATH "/Bootstrap.php"
bootstrap.class = "Bootstrap"
#配置工程模块
resources.frontController.moduleDirectory = APPLICATION_PATH "/modules"
```

配置文件存放位置和名称并不固定，只要与实例 Zend_Application 类传递的参数一致即可。

注意 这里配置的是多个工程模块，moduleDirectory=APPLICATION_PATH "/modules"，默认控制器放置在 default 文件夹下，文件夹架构如图 25.5 所示。如果配置的是单个工程模块，controllerDirectory= APPLICATION_PATH "/controllers"，文件夹架构如图 25.2 所示。

（4）在 application.ini 配置文件中指定启动类的位置后，工程就可以找到启动文件 Bootstrap.php。其关键代码如下：

```
class Bootstrap extends Zend_Application_Bootstrap_Bootstrap{
//基本的启动类可以不进行任何操作
}
```

从上述代码可知，启动类 Bootstrap 继承自类 Zend_Application_Bootstrap_Bootstrap，该类中已经实现最基本的启动配置工作，所以在启动类中不需包含任何代码就可以运行最基本的 Zend Framework 应用。

（5）完成以上步骤后，就可以创建默认控制器 IndexController，同样一个最基本的控制器应该包含一个首页动作 indexAction。其代码如下：

```
class IndexController extends Zend_Controller_Action{
    public function indexAction (){                        //默认动作
        $this->view->testStr="Hello Zend Framework!";      //为视图变量赋值
    }
}
```

说明 使用 Zend Framework 时，控制器名应该使用 Controller 结束，动作名应以 Action 结束。视图文件夹和控制器文件夹必须在同一目录下，这里都存储在 application\modules\default 文件夹下。

（6）完成默认控制器及首页动作的创建后，就可以创建视图文件 index.phtml，为了查看效果，在视图文件中仅输出首页动作中指定的视图变量的值。其代码如下：

```
echo $this->testStr;
```

至此，多模块 MVC 框架结构创建完成，其完整的文件夹架构如图 25.5 所示。

假设上述工程所在目录为 Apache 默认主目录的 TM\25\25.1 子目录下，则可以通过如下 URL 进行访问。

☑ http://127.0.0.1:82/TM/25/25.1/public/。
☑ http://localhost:82/TM/25/25.1/public/index。
☑ http://localhost:82/TM/25/25.1/public/index/index。

其运行效果如图 25.6 所示。



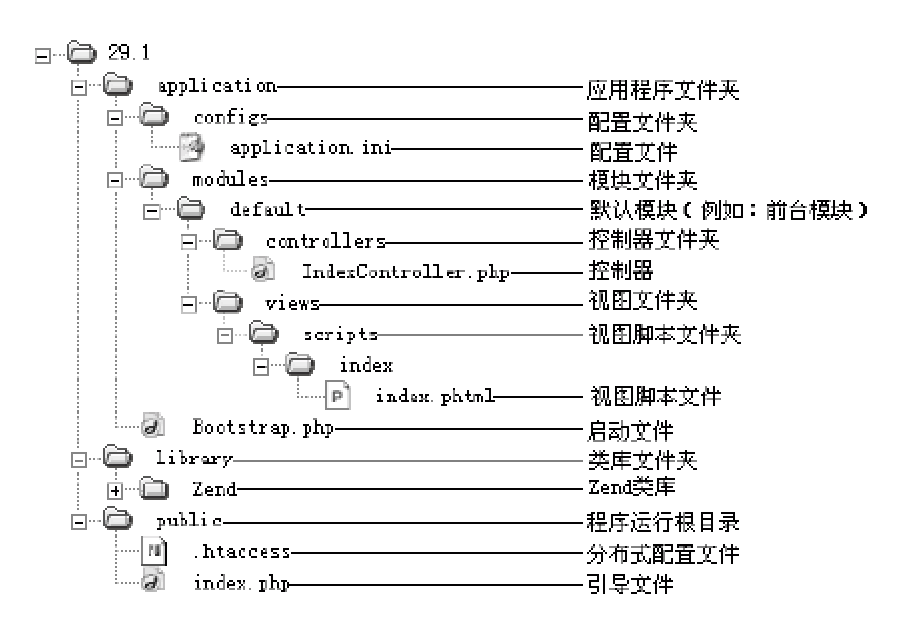



图 25.5　多模块 MVC 框架结构



图 25.6　Zend Framework 框架创建演示

正常情况下，Zend Framework 的访问 URL 应该为“域名+模块名+控制器名+动作名”，那么为什么上述 URL 省略了部分内容也可以访问呢？这是因为在 Zend Framework 中，默认模块名、默认控制器名和首页动作名可以省略。

25.2.4　Zend Framework 的编码标准

程序人员必须养成良好的编码习惯，为编码打下一个良好的基础。在每一个项目中都会存在一种编码标准，这种编码标准会提高程序质量，减少 BUG，使维护和二次开发变得轻松。Zend Framework 编码标准是针对 Zend Framework 框架开发所设计的，它可以为参与 Zend Framework 开发的个人和团队所使用，也可以在没有特定编码标准的项目中使用。Zend Framework 编码标准如下。

1. 命名约定

☑ 类：类的名称只允许使用字母和数字，这里不鼓励使用数字，如果类名称中包含多个单词，则每个单词的第一个字母必须大写。其中下划线只被允许分隔路径。例如，Zend/Application.php 文件中对应的类名称应该为 Zend_Application。

☑ 文件名：文件名称只允许使用字母、数字、下划线和短横线（-），不可使用空格。同时，任何存在 PHP 代码的文件都必须以.php 扩展名结尾（视图文件除外）。

☑ 函数和方法：函数名称只允许使用字母和数字，不允许使用下划线。这里同样不鼓励使用数字。函数的名称总是以小写开始，当包含多个单词时，每个单词的第一个字母大写。在对象方法中被定义为 private 或者 protected 的，名称必须以一个下划线开始。

☑ 变量：变量名称只允许使用字母和数字，不允许使用下划线。这里同样不鼓励使用数字。变量的名称总是以小写开始，当包含多个单词时，每个单词的第一个字母大写。在对象方法中被定义为 private 或者 protected 的，名称必须以一个下划线开始。

☑ 常量：常量名称只允许使用字母、数字和下划线。常量名称所有字母都必须为大写。每个单词都必须以下划线分隔。在对象方法中定义常量必须通过 const 定义类的成员，这里不鼓励使用 define 定义常量。

注意　除了类名称之外，其余命名都鼓励使用长字符，并尽量采用能够说明意图和行为的单词。

2. 编码风格

☑ 缩进：缩进由 4 个空格组成。需要注意的是，不允许使用 Tab 键。

☑ 行长度：一行代码的最佳长度在 80 个字符以内。如果情况特殊，最多可以延长到 120 个字符。

☑ 数组：当使用数组时，每个逗号后都用一个空格来提高可读性。如果是关联数组，则把代码分为

多行，在每一个连续的开头用空格补充并对齐键和值。例如：

```
$Array = array( 'oneKey'    => 'oneValue',
                'twoKey' => 'twoValue');
```

3．注释文档

无论是在类中，还是在包含 PHP 代码的文件中，必须至少在文件顶部包含一些注释信息，对类或者 PHP 代码进行解释说明。其代码如下：

```
/**
* 文件/类的简短描述
* 文件/类的详细描述（如果有的话）… …
*/
```

函数和方法信息则必须在封装的函数上方给出注释信息。代码如下：

```
/**
* 函数的描述
* 所有参数
* 所有可能的返回值（类型）
* @return void
* 如果函数/方法抛出异常，使用@throws 于所有已经知道的异常类中
* @throws Zend_Application_Exception
*/
```

25.3　Zend_Auth 身份认证

Zend_Auth 是登录模块中对身份进行验证的组件，但是并不是授权。Zend_Auth 具体作用是定义一些证书来确定身份是否是 Auth 声明的。其中这些“证书”被叫做 Zend_Auth 适配器。

25.3.1　Zend_Auth 适配器

Zend_Auth 适配器被用来认证特定的服务器。例如接受认证证书，认证后返回的结果对于 Zend_Auth 适配器都是通用的。身份认证必须符合程序要求，这就需要编写适合程序的适配器类。

Zend_Auth 适配器类必须实现 Zend_Auth_Adapter_Interface 类，调用接口函数 authenticate()，执行认证查询。例如，简单介绍一个身份认证适配器类，仅仅针对固定用户名和密码。程序代码如下：

```
<?php
/**
 * 认证用户名和密码的适配器
 */
class Model_AuthAdapter implements Zend_Auth_Adapter_Interface{
    protected $_username;
    protected $_password;
    /**
     * Construct
     *
     * 带入用户名和密码
     *
     * @return void
     */
    public function __construct($username,$password){
        $this->_username = $username;
```

```
        $this->_password = $password;
    }
    /**
     * 对用户名和密码进行身份认证
     *
     * @throws Zend_Auth_Adapter_Exception If authentication cannot
     *                                     be performed
     * @return Zend_Auth_Result
     */
    public function authenticate(){
        $array = array();
        if (($this->_username == 'mr') && (($this->_password == 'mrsoft'))){
            $array[0] = true;
❶          return new Zend_Auth_Result(1,$array);
        }else{
            $array[0] = false;
            return new Zend_Auth_Result(-1,$array);
        }
    }
}
```

技巧 ❶ Zend_Auth_Result 方法：将参数带入 Zend_Auth_Result 对象，验证时只需调用 Zend_Auth_Result 对象中的 isValid()方法。

Zend_Auth 适配器返回一个带有 authenticate()的 Zend_Auth_Result 实例。在 Zend_Auth_Result 对象中，提供如下 4 个方法对 Zend_Auth 适配器返回结果进行操作。

- ☑ isValid()：返回 true 当且仅当结果表示一个成功的认证尝试。
- ☑ getCode()：返回一个 Zend_Auth_Result。常量标识符用来决定认证失败的类型或者是否认证成功。可以用于开发者希望区别若干认证结果类型的情形。
- ☑ getIdentity()：返回认证尝试的身份。
- ☑ getMessages()：返回认证尝试失败的数组。

25.3.2　身份持久认证

身份的持久认证必须结合 Zend_Session 组件，Zend_Session 组件通过会话（session）实现服务器端和客户端之间的持久联系。Zend_Session 采用 Zend_Session_Namespace 对象的方法管理，而且会话空间（Session Namespaces）提供了命名空间来管理会话数据。这样做的好处是可以在不同页面间调用不同的 Session 会话空间。

Zend_Session 主要使用方法如下：

```
<?php
include 'Zend/Session/Namespace.php';
$nameSpace = new Zend_Session_Namespace('Namespace');                    //开启 Session
```

下面通过实例讲解如何使用 Zend_Auth 和 Zend_Session 对身份持久认证。

例 25.2　结合应用 Zend_Auth 和 Zend_Session 来对身份持久认证。**（实例位置：光盘\TM\Instance\25\25.2）**

（1）将 Zend_Auth 适配器类带入本实例中，放入 models 文件夹。根据 Zend Framework 编码标准重新编写适配器类名称。代码如下：

```
class Model_AuthAdapter implements Zend_Auth_Adapter_Interface{
    protected $_username;                                                //声明成员变量
protected $_password;                                                    //声明成员变量
```

```
public function __construct($username,$password){                            //创建构造函数
        $this->_username = $username;                                        //为成员变量赋值
        $this->_password = $password;                                        //为成员变量赋值
}
public function authenticate(){                                              //定义方法对身份进行认证
        $array = array();
        if (($this->_username == 'mr') && (($this->_password == 'mrsoft'))){ //判断用户名和密码
            $array[0] = true;
            return new Zend_Auth_Result(1,$array);                           //正确返回结果
        }else{
            $array[0] = false;
            return new Zend_Auth_Result(-1,$array);                          //错误返回结果
        }
}
}
```

说明 在 Model 模型层写入的类可以直接在程序中实例化，而不需要引用文件。

（2）控制器中功能分为验证用户名和密码、成功页面输出和安全退出。3 个功能分别对应 3 个方法。其程序代码如下：

```
<?php
class IndexController extends Zend_Controller_Action{
public function indexAction(){
$this->view->assign("title","网站登录界面");
❶       if($this->_request->isPost()){                                       //是否 POST 传输
$username = $this->_request->getPost('username');                            //取得 username 值
$password = $this->_request->getPost('password');                            //取得 password 值
$auth = Zend_Auth::getInstance();                                            //调用 Zend_Auth
$authModel = new Model_AuthAdapter($username, $password);
//将适配器结果放入 Zend_Auth 的结果集中
❷           $result = $auth->authenticate($authModel);
if($result->isValid()){                                                      //使用默认验证
//开启 SESSION
$sessionNameSpace = new Zend_Session_Namespace('project');
$sessionNameSpace->username = $username;                                     //赋予 session 值
$sessionNameSpace->password = $password;
$this->_redirect('index/success');                                           //跳转到成功页面
}else {
echo "用户名和密码错误";
}
}
}
public function successAction(){
$this->view->assign('title', '成功界面');
$sessionNameSpace = new Zend_Session_Namespace('project');
//判断 session 中是否设定 username 变量，即是否登录
❸       if($sessionNameSpace->username == ""){
die("不允许直接访问此页面");
}
$this->view->username = $sessionNameSpace->username;
$this->view->password = $sessionNameSpace->password;
}
❹   public function logoutAction(){
```

```
$sessionNameSpace = new Zend_Session_Namespace('project');
unset($sessionNameSpace->username);                              //销毁 username 变量
$this->_redirect('index/index');
}
}
```

说明

❶ $this->_request->isPost()方法：判断当前页面是否使用 POST 方式提交。

❷ authenticate()方法：适配器类中调用 Zend_Auth_Result 对象方法。

❸ 加入不允许未登录访问本页面的功能。通过 session 判断。

❹ 安全退出方法：销毁 session 变量。

（3）从控制器中可以看出存在 3 个页面：index、success 和 logout。但是仔细推敲后发现真正输出数据的页面只有 index 和 success，logout 页面没有任何数据输出。所以视图层中只建立 index.phtml 和 success.phtml 两个页面。index.phtml 页面关键代码如下：

```
❶<title><?php echo $this->escape($this->title); ?></title>
<style type="text/css">
.submit {
	font-weight:bold;
	width:112px;
	border:medium none;
❷	background:#FFF url(<?php echo $this->baseUrl("images/index_02.jpg"); ?>) no-repeat;
	cursor:pointer;
	line-height:33px;
	font-size:14px;
}
</style>
❸<form id="form1" name="form1" method="post" action="<?php echo $this->Url(array('controller'=>'index',
'action'=>'index')); ?>">
<table width="338" height="168" background="<?php echo $this->baseUrl("images/index_01.jpg"); ?>">
    <tr>
        <td height="22"><input name="username" type="text" size="24" maxlength="10" /></td>
    </tr>
    <tr>
        <td><input name="password" type="password" size="25" maxlength="10" /></td>
    </tr>
</table>
<table width="338" height="62" background="<?php echo $this->baseUrl("images/index_04.jpg"); ?>">
    <tr>
        <td><input type="submit" name="submit" class="submit" value="    "/></td>
        <td><img src="<?php echo $this->baseUrl("images/index_03.jpg"); ?>" border="0"/></td>
    </tr>
</table>
</form>
```

说明

❶ $this->escape($this->title)：输出标题。

❷ $this->baseUrl("images/index_02.jpg")：设置背景颜色。

❸ $this->Url(array('controller'=>'index','action'=>'index'))：设置表单中数据提交的路径。

在 form 表单中的 action 表示提交数据到哪一个程序。Zend Framework 框架中，如果提交的数据不是当前控制器，必须使用$this->Url()方法确认控制器。

success.phtml 页面代码如下：

```
<title><?php echo $this->escape($this->title); ?></title>
<table width="300" border="1" cellpadding="0" cellspacing="0" bordercolor="#333333">
  <tr height="30">
    <td width="60" align="center">用户名：</td>
    <td><?php echo $this->escape($this->username); ?></td>
  </tr>
  <tr height="30">
    <td align="center">密  码：</td>
    <td><?php echo $this->escape($this->password); ?></td>
  </tr>
  <tr height="30">
    <td><a href="<?php echo $this->Url(array('controller'=>'index','action'=>'logout')); ?>">安全退出</a></td>
  </tr>
</table>
```

技巧 在视图层中，如果要调用图片或者各种文件，可以使用 Zend Framework 中的 baseUrl()函数，这样调用的位置就是入口文件所在文件夹。

其运行结果如图 25.7 所示。



图 25.7　身份的持久认证

25.3.3　数据库认证

Zend Framework 框架中通过 Zend_Auth_Adapter_DbTable()组件实现数据库认证。

首先，应用 Zend_Db_Adapter_Abstract 连接一个数据库，并将返回对象传递给 Zend_Auth_Adapter_DbTable 的构造器。

然后，实例化 Zend_Auth_Adapter_DbTable，并通过配置选项传递参数，以及用户登录的用户名和密码，实现数据库的认证。

可用的配置选项如下。

☑ tableName：包含认证证书的数据库表名，执行数据库认证查询需要依靠该证书。

☑ identityColumn：数据库表的列名称，用来表示身份。身份列必须包含唯一的值，例如用户名或者 E-mail 地址。

☑ credentialColumn：数据库表的列名称，用来表示证书。在一个简单的身份和密码认证 scheme 下，证书的值对应为密码。

☑ credentialTreatment：在许多情况下，密码和其他敏感数据是加密的。通过指定参数化的字串来使用这个方法，如'MD5(?)' 或者 'PASSWORD(?)'，开发者可以在输入证书数据时使用任意的 SQL 语句。



最后，调用 authenticate()方法执行认证查询。在认证结果对象中，可以通过 getIdentity()方法获取认证的身份，也可以通过 getResultRowObject()方法返回一行数据。

25.4　Zend_Db 数据库操作

Web 语言编程离不开数据库的支持。Zend Framework 框架中连接和操作数据库的是 Zend_Db 组件。Zend_Db 组件是一个基于 PDO 模块的数据库抽象层 API，它可以支持多种数据库，如 MySQL、SQLite、SQL server 等。

25.4.1　Zend_Db_Adapter 数据库操作

应用 Zend_Db_Adapter 必须静态调用 Zend_Db::factory()方法。例如，调用 PHP 的最佳搭档 MySQL 数据库。程序代码如下：

```
<?php
include_once 'Zend/Db.php';                              //包含 Zend_Db 文件
$params = array(
            'host'=> 'localhost',                        //连接数据库 URL
            'username'=>'root',                          //连接数据库用户名
            'password'=>'111',                           //连接数据库密码
            'dbname'  =>'db_database25',                 //连接数据库名称
);
$db = Zend_Db::factory('PDO_MYSQL', $params);            //基于 PDO 连接 MySQL 数据库
```

这就是一个简单的连接和操作数据库的方法。但是由于 Zend Framework 的版本升级到 1.8 以上，入口文件中并不需要调用此方法，所以此方法已经不能够满足现在的需求，当然并不是不可以使用。下面介绍的 Zend_Db_Table 是 Zend Framework 推荐的连接和操作数据库的组件。

25.4.2　Zend_Db_Table 数据库操作

Zend_Db_Table 是 Zend Framework 的表模块。它通过 zend_db_adapter 连接到数据库，创建类来继承 Zend_Db_Table 类，实例化创建的类，通过返回的对象对数据表进行操作和查询。

例 25.3　在 Zend Framework 1.8 及以上版本中连接和操作 MySQL 数据库，查询表 tb_user 中数据，并输出查询结果。**（实例位置：光盘\TM\Instance\25\25.3）**

（1）搭建 Zend Framework 的基本环境。

（2）使用 MySQL 数据库，根据 Zend Framework 1.8 以上版本资源加载方法（详细情况查看 25.2.3 基本环境搭建中的启动文件），将连接数据库信息写入配置文件。程序代码如下：

```
resources.db.adapter = "PDO_MYSQL"                               //基于 PDO_MYSQL 连接数据库
resources.db.params.host = "localhost"                           //数据库 URL
resources.db.params.username = "root"                            //数据库用户名
resources.db.params.password = "111"                             //数据库密码
resources.db.params.dbname = "db_database25"                     //数据库名称
resources.db.params.driver_options.1002 = "set names utf8"       //数据库字符集
```

（3）在启动文件中，通过_init 方法加载数据库资源。程序代码如下：

```
protected function _initDB(){
❶    $options = $this->getOption('resources');
```

```
❷      $options = $options['db'];
❸      $resources = $this->getPluginResource('db');
❹      $db = $resources->getDbAdapter();
       Zend_Db_Table::setDefaultAdapter($db);
       Zend_Registry::set('dbAdapter',$db);
       Zend_Registry::set('dbprefix',$options['params']['prefix']);
}
```

说明

❶ getOption()方法：在配置文件中找到 resources 开头的资源配置。

❷ $options['db']：获取数据库资源，在资源配置中找到数据库（db）项。

❸ getPluginResource()方法：获取系统中数据库资源。

❹ $resources->getDbAdapter()：定义数据库操作变量。在程序中只要通过调用 Zend_Registry::get 方法调用此变量，即可实现对数据库的操作。

（4）通过 Zend_Registry::get 方法调用数据库资源变量，通过 fetchAll 方法执行查询语句，最后通过 print_r 输出查询结果。程序代码如下：

```
public function indexAction(){
        //调用 db 对象，dbAdapter 为启动文件中设置调用资源
        $this->_db = Zend_Registry::get('dbAdapter');
        $sql = "select * from tb_user";                  //SQL 语句
        $result = $this->_db->fetchAll($sql);            //得到 SQL 语句结果，查看 fetchAll
        print_r($result);
}
```

运行本实例，查看数据表中信息，会抛出控制器错误，如图 25.8 所示。

这是因为 Zend Framework 框架找不到控制器（IndexController）对应的视图文件，所以需要在视图文件夹（views）下创建一个 index.phtml 视图页面，在 index.phtml 页面中可以不加载任何程序。本实例的运行效果如图 25.9 所示。

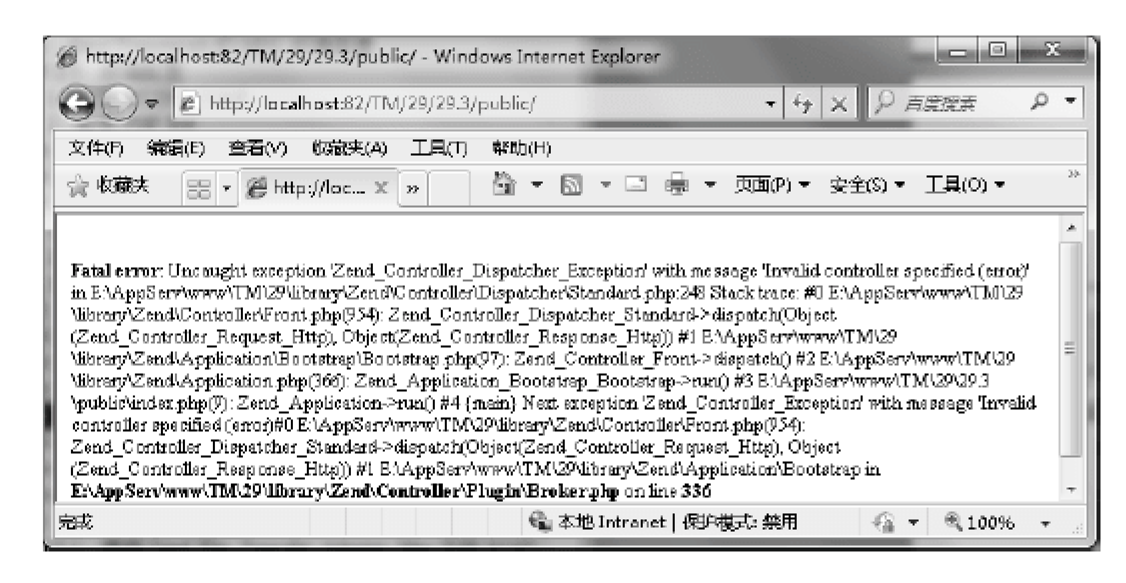


图 25.8　错误显示

```
Array ( [0] => Array ( [id] => 1 [netname] => mr [password] => fdb390e945559e74475ed8c8bbb48ca5 [email] =>
mingrisoft@mingrisoft.com [fromcity] => 吉林-长春 [regip] => 127.0.0.1 [lastlogintime] => 2010-05-07 09:16:43 [lastloginip] =>
127.0.0.1 [regtime] => 2010-05-07 09:16:43 [usertype] => 1 [score] => 0 [face] => [signature] => [friendid] => [spacebrowse] => 0 ) [1]
=> Array ( [id] => 103 [netname] => mingri [password] => mrsoft [email] => [fromcity] => [regip] => [lastlogintime] => [lastloginip]
=> [regtime] => [usertype] => [score] => [face] => [signature] => [friendid] => [spacebrowse] => ) [2] => Array ( [id] => 105
[netname] => mingri [password] => mrsoft [email] => [fromcity] => [regip] => [lastlogintime] => [lastloginip] => [regtime] =>
[usertype] => [score] => [face] => [signature] => [friendid] => [spacebrowse] => ) )
```

图 25.9　运行效果图

25.4.3　数据表类

通过继承 Zend_Db_Table 类，可以完成对数据表的操作，而对数据表的操作多数都是代码的重用，所以最好的方法就是将其定义到模型中，这样可以提高代码的重用率，同时也便于对程序进行更新和维护。

通过模型类继承 Zend_Db_Table 类，封装对一个数据库的操作方法是非常简单的。数据表类中对数据库的操作主要分为 4 种。

☑ insert($dataArray)：添加数据，参数$dataArray 是对应数据表的联合数组。

☑ update($setArray,$where)：修改数据，参数$setArray 是对应数据表的联合数组，参数$where 是修改



数据的条件，相当于 where 语句。

☑ delete($where)：删除数据，参数$where 是删除数据的条件，相当于 where 语句。

☑ fetchAll($where = null, $order = null, $count = null, $offset = null)：查询数据。如果是单纯的查询操作可以直接使用 fetchAll()方法。但是，当查询有很多附加条件时，就要使用 fetchAll()方法的参数，其参数说明如下。

  - $where：表示 where 语句。
  - $order：表示 order by 语句。
  - $count：表示 limit 语句中显示多少结果行。
  - $offset：表示 limit 语句中从第几个数据开始。

例 25.4　创建数据表类，通过数据表类完成对数据库的添加、删除、修改和查询。**（实例位置：光盘\TM\Instance\25\25.4）**

（1）在 public 目录下建立 URL 重写文件.htaccess，并在该文件中输入 URL 重写规则。代码如下：

```
#开启 URL 重写
RewriteEngine on
#除扩展名为.js、.css、.gif、.jpg、.png、.bmp 的文件外，访问其他文件都转向到 index.php 引导文件
RewriteRule !\.(js|css|gif|jpg|png|bmp)$ index.php
```

（2）在 public 目录下创建 index.php 引导文件。代码如下：

```
defined('APPLICATION_PATH') || define('APPLICATION_PATH', realpath(dirname(__FILE__) . '/../application'));
                                                                    //应用路径
defined('APPLICATION_ENV')  ||  define('APPLICATION_ENV',  getenv('APPLICATION_ENV')  ?  getenv
('APPLICATION_ENV') : 'project');                                   //应用环境
$arrayIncludePath = array('.', realpath(dirname(__FILE__) . '/../../../library'));    //指定工程包含目录
set_include_path(implode(PATH_SEPARATOR, $arrayIncludePath));        //将指定路径包含到工程中
require_once 'Zend/Application.php';                                 //包含 Application.php 文件
$application = new Zend_Application(APPLICATION_ENV, APPLICATION_PATH . '/configs/application.ini');
                                                                    //实例 Zend_Application 类
$application->bootstrap()->run();                                   //调用启动文件并运行项目
```

首先，定义常量 APPLICATION_PATH 和 APPLICATION_ENV，分别用来保存应用路径和应用环境名。然后，将当前目录地址“.”和 Zend Framework 类库地址作为数组元素保存在名为$arrayIncludePath 的数组中，并使用 implode()函数将上述路径用 PATH_SEPARATOR 常量连接。最后，使用 set_include_path()方法将路径导入到工程中，这样工程就可以直接找到这些路径下的文件。

Zend Framework 类库被导入到工程后，使用 require_once 包含语句包含 Zend 目录下的 Application.php 文件，之后实例化 Zend_Application 类，并通过引用该类的 bootstrap()方法调用启动类，再调用 run()方法运行工程。

（3）在第（2）步实例化 Zend_Application 类时，为类的构造函数传入一个名为 application.ini 的文件，即工程的配置文件。其代码如下：

```
[project]
#设置错误级别
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1
#设置时区
phpSettings.date.timezone = 'Asia/Shanghai'
#配置启动类
bootstrap.path = APPLICATION_PATH "/Bootstrap.php"
bootstrap.class = "Bootstrap"
#配置工程模块
resources.frontController.controllerDirectory = APPLICATION_PATH "/controllers"
#配置数据库 PDO 模块
```

```
resources.db.adapter = "PDO_MYSQL"
#设置数据库服务器
resources.db.params.host = "localhost"
#设置服务器用户名
resources.db.params.username = "root"
#设置服务器密码
resources.db.params.password = "111"
#设置数据库
resources.db.params.dbname = "db_database25"
#设置数据库编码格式
resources.db.params.driver_options.1002 = "set names utf8"
```

（4）在 application.ini 配置文件中指定启动类的位置后，工程就可以找到启动文件 Bootstrap.php。其关键代码如下：

```
class Bootstrap extends Zend_Application_Bootstrap_Bootstrap{
public function _initAutoload(){
$moduleAutoloader = new Zend_Application_Module_Autoloader(array('namespace' => '','basePath' => '../application'));
return $moduleAutoloader;
}
}
```

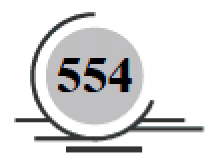
启动类 Bootstrap 继承自类 Zend_Application_Bootstrap_Bootstrap，该类中已经实现最基本的启动配置工作。

（5）创建模型 models，创建数据表类 User，定义操作数据表的方法。其代码如下：

```
<?php
class Model_User extends Zend_Db_Table{                //定义数据表类，继承 Zend_Db_Table
    protected $_name = "tb_user";                      //定义数据表名称变量
    protected $_primary = "id";                        //定义数据 id 变量
    protected $_adapter;                               //定义数据库连接标识变量
    public function init(){                            //定义 init()方法
        $this->_adapter = $this->getAdapter();         //获取操作数据表的对象
    }
}
```

（6）创建控制器 controllers、默认控制器 IndexController，定义添加、修改、删除和查询动作的方法，实例化数据表（tb_user）模型中定义的 User 类，通过返回的对象调用数据库操作方法，完成对数据库的操作。其关键代码如下：

```
<?php
class IndexController extends Zend_Controller_Action{
    public function indexAction(){
        $table = new Model_User();                     //类的实例化
        //增加的数据
        $data = array(
            'netname' => 'King',
            'password'   => 'mrsoft',
        );
❶       if($table->insert($data)){                     //执行添加操作
            $this->view->insert = "插入数据成功！ ";
        }
    }
    public function deleteAction(){                    //定义删除方法
        $table = new Model_User();                     //类的实例化
        $where = 'id = 2';                             //定义删除的条件
❷       if($table->delete($where)){                    //执行删除操作
            $this->view->delete = "删除成功！ ";
```



```
            }else {
                $this->view->delete = "该数据已经不存在";
            }
        }
        public function updateAction(){                                      //定义更新方法
            $table = new Model_User();                                       //类的实例化
            $set = array(
                'netname' => 'mingri',
            );
            $where = 'id = 2';                                               //定义更新条件
❸           if($table->update($set, $where)){                               //执行更新操作
                $this->view->update = "修改成功！ ";
            }else {
                $this->view->update = "该数据不存在或已经被修改过";
            }
        }
        public function fetchAction(){                                       //定义查询方法
            $table = new Model_User();                                       //类的实例化
            //定义查询条件
            $where = null;
            $order = "id desc";                                              //以 ID 降序排列
            $count = 3;                                                      //输出 3 条记录
            $offset = 0;                                                     //从第 1 条记录开始
❹           $result_all = $table->fetchAll($where, $order, $count, $offset); //执行查询语句
            $this->view->select =$result_all;
        }
}
```

说明

❶ insert()方法：相当于执行 insert into tb_user(login_name,login_pwd) values('king', 'mrsoft')语句。

❷ delete()方法：相当于执行 delete from tb_user where id = 2 语句。

❸ update()方法：相当于执行 update tb_user set login_name = 'mingri' where id = 2 语句。

❹ fetchAll()：可以无参数直接使用，但是当使用参数时，参数位置必须对应。

（7）控制器和动作方法创建完成后，创建与控制器对应的视图文件，这里创建 4 个视图文件：index.phtml、update.phtml、delete.phtml 和 fetch.phtml，输出对数据库的操作结果。在 fetch.phtml 视图文件中，通过 foreach 语句循环输出数据库的查询结果。其代码如下：

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>查看数据</title>
<link href="<?php echo $this->baseUrl('css/style.css');?>" rel="stylesheet" type="text/css" />
</head>
<body>
<table width="600" border="1" cellpadding="1" cellspacing="1" bordercolor="#FFFFFF" bgcolor="#336600">
    <tr>
        <td height="25" align="center" bgcolor="#FFFFFF">ID</td
        <td align="center" bgcolor="#FFFFFF">用户名</td>
        <td align="center" bgcolor="#FFFFFF">密码</td>
    </tr>
    <?php foreach ($this->select as $key => $select){?>
    <tr>
```

```
            <td height="20" align="center" bgcolor="#FFFFFF"><?php echo $select['id'];?></td>
            <td align="center" bgcolor="#FFFFFF"><?php echo $select['login_name'];?></td>
            <td align="center" bgcolor="#FFFFFF"><?php echo $select['login_pwd'];?></td>
        </tr>
        <?php }?>
        <tr>
            <td height="30" colspan="3" align="center" bgcolor="#FFFFFF">
            <table width="300" border="0" cellpadding="0" cellspacing="0">
                <tr>
                    <td><a href="index">添加数据</a></td>
                    <td><a href="update">更新</a></td>
                    <td><a href="delete">删除</a></td>
                    <td height="28"><a href="fetch">查看</a></td>
                </tr>
            </table>
            </td>
        </tr>
</table>
</body>
</html>
```

运行结果如图 25.10 所示。

ID	用户名	密码
58	King	mrsoft
57	King	mrsoft
56	King	mrsoft
添加数据　更新　删除　查看		

图 25.10　查看数据库中数据

25.5　Zend_File 文件控制

Zend_File 组件主要用于实现文件的上传和下载，该组件设计上采用适配器方式，这样无论使用 HTTP 的 POST 方式还是使用 FTP 方式对文件进行上传，其具体实现代码都相同，如果上传方式发生改变，只需更改适配器即可。

由于该组件尚不成熟，目前只支持 HTTP 的 POST 方式上传，所以只能使用 Zend_File_Transfer_Adapter_Http 适配器实现文件的上传，但 Zend Framework 开发团队不久将推出更多的适配器（如支持 FTP 方式的适配器）。本节将主要介绍 Zend_File_Transfer_Adapter_Http 的使用方法。

25.5.1　使用 Zend_File_Transfer_Adapter_Http 实现 POST 方式文件上传

使用 Zend_File_Transfer_Adapter_Http 实现文件上传主要使用该类的 setDestination()和 receive()方法实现，其中 setDestination()方法用于指定上传文件保存在服务器端的目录，receive()方法用于执行上传操作。下面将通过一个简单的实例讲解使用该适配器类实现 POST 方式文件上传的方法。

例 25.5　使用 Zend_File_Transfer_Adapter_Http 适配器类实现文件上传。（**实例位置：光盘\TM\Instance\25\25.5**）

为了能够掌握使用上述适配器类实现文件上传的最核心方法，实现本实例时只建立一个文件上传表单和一个用于实现上传业务逻辑的控制器类。



（1）建立 Zend Framework 的 MVC 框架结构，具体创建过程请参见本章第 25.2 节的内容，这里不再赘述。

（2）建立用于实现文件上传的表单。为了便于查看上传效果，在该表单中只设置用于选择上传文件的文件域和用于提交表单的图片按钮。其代码如下：

```
<form enctype="multipart/form-data" action="<?php echo $this->Url(array('controller'=>'index','action'=>'index')); ?>"
method="POST">
    <span class="STYLE1">选择上传文件:</span>
<input name="uploadedfile" type="file" />  
<input type="image" name="imageField" src="<?php echo $this->baseUrl('images/sc.jpg');?>" />
</form>
```

通过 POST 方法实现文件上传的表单 form 标签中必须设置 enctype 属性的值为 multipart/form-data，并且 method 属性的值必须为 POST。

（3）建立上传表单后，就可以编写用于实现文件的业务逻辑。本实例实现文件上传的代码是在 IndexController 控制器类的 indexAction 动作中。其关键代码如下：

```
<?php
class IndexController extends Zend_Controller_Action{
    public function indexAction(){
        if($this->_request->isPost()){                              //判断是否是 POST 上传
            $adapter = new Zend_File_Transfer_Adapter_Http();
            $adapter->setDestination('./upfiles');                  //设置上传文件存储路径
            if(!$adapter->receive()){                               //执行上传操作
                $messages = $adapter->getMessages();                //获取返回的错误信息
                echo implode("<br>",$messages);                     //输出错误信息
            }else{
                echo "<script>alert('上传成功！');</script>";
            }
        }
    }
}
```

在上述代码中，首先使用当前控制器对象的_request 属性的 isPost()方法判断是否已经提交了表单，如果是则进行文件上传操作。

在具体实现文件上传时，首先通过 new 关键字实例化 Zend_File_Adapter_Http()适配器类，然后调用该适配器类的 setDestination()方法指定上传文件要保存的服务器端目录，最后通过调用该适配器类的 receive()方法实现文件上传。这里需要注意，如果上传失败，则调用 receive()方法后将返回 false 值，并可以通过上述适配器类的 getMessages()方法返回上传失败的原因。

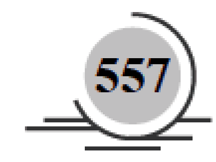
运行上述实例，如图 25.11 所示，在表单的文件域中选择要上传的文件，然后单击“上传”按钮，则所上传的文件将被保存到该实例主目录的 upfiles 子目录中。



图 25.11　Zend_File_Transfer_Adapter_Http 实现 POST 方式文件上传

根据上述实例可知，通过 Zend_File_Transfer_Adapter_Http 适配器类实现文件上传的关键代码只有如下 3 行：

```
$adapter = new Zend_File_Transfer_Adapter_Http();                   //实例适配器类
$adapter->setDestination('./upfiles');                              //定义上传文件保存路径
$adapter->receive();                                                //执行上传操作
```



25.5.2 对上传文件的合理性验证

上传文件时，有时需要对文件的类型、大小进行限制。例如使用 PHP 开发的 Web 应用在制作文件上传模块时，为了提高项目的安全性，不允许上传扩展名为.php 或.exe 的文件，那么使用 Zend_File_Transfer_Adapter_Http 适配器类如何实现上传文件的有效性验证呢？

在具体讲解使用 Zend_File_Transfer_Adapter_Http 适配器类实现文件上传的合理性验证前，首先应该知道常用的验证规则有哪些，其中表 25.2 所示为该适配器类常用的效验规则。

表 25.2 Zend_File_Transfer_Adapter_Http 适配器类常用效验规则

效 验 规 则	说 明
Count	指定上传文件的数量范围验证规则
ExcludeExtension	指定不允许的上传文件扩展名验证规则
Extension	指定允许的上传文件扩展名验证规则
ImageSize	指定上传图片大小的验证规则
IsImage	指定上传文件是否为图片的验证规则
size	指定上传文件大小的验证规则
ExcludeMimeType	指定上传文件不允许的 mime 类型验证规则
MimeType	指定上传文件允许的 mime 类型验证规则
Exists	指定要上传的文件已经存在于服务器的验证规则
NotExists	指定要上传的文件不存在于服务器的验证规则

了解表 25.2 中的常用验证规则后，又如何将这些规则应用于上传过程中呢？这里只需使用上传适配器类的 addValidator()方法在上传过程中指定上传规则即可，该方法一般包含 3 个参数，第 1 个参数用于指定表 25.2 中的验证规则名称；第 2 个参数为 boolean 型，如果为 true 则当该条验证失败时继续向下验证，否则不再继续向下验证；第 3 个参数为数组型，用于设置具体验证方式和验证错误的提示信息。

例如，在图片上传时使用 Zend_File_Transfer_Adapter_Http 适配器类验证图片大小只能小于 1MB 的代码如下。

```
$adapter = new Zend_File_Transfer_Adapter_Http();                    //实例化适配器类
$adapter->setDestination('./upfiles');                               //定义上传文件保存路径
$adapter->addValidator('Size', false, array('min' => '0kB' , 'max' => '1MB' , 'bytestring' => false , 'messages' =>
'・您所上传的图片不能超过 1MB'));                                    //对文件大小进行验证
$adapter->receive();                                                 //执行上传操作
```

25.5.3 为上传增加过滤规则

通过适配器类的 addValidator()方法可以对上传文件进行有效验证，从而可以有效地提高系统的安全性和人性化程度，但在实际应用中，保存上传文件时需要更改上传文件名或对上传文件进行加密操作，这时就需要使用适配器类的 addFilter()方法为上传增加过滤规则。在具体讲解 addFilter()方法前，首先需要了解常用的过滤规则有哪些，其中表 25.3 所示为 Zend_File_Transfer_Adapter_Http 适配器类常用的过滤规则。

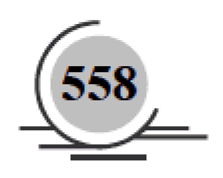


表 25.3　Zend_File_Transfer_Adapter_Http 适配器类常用的过滤规则

过 滤 规 则	说　　明
Decrypt	对上传文件进行解密过滤
Encrypt	对上传文件进行加密过滤
LowerCase	将上传的文本类型的文件中的英文字符转换为小写
UpperCase	将上传的文本类型的文件中的英文字符转换为大写
Rename	对上传文件进行重命名

使用 addFilter()方法为上传文件增加验证规则一般需要为该方法指定 3 个参数，第 1 个参数为要指定的过滤规则名称；第 2 个参数为数组型，用于为该过滤指定具体的过滤规则；第 3 个参数用来指定该过滤规则要限定的文件域名称。

例如，在图片上传时使用 Zend_File_Transfer_Adapter_Http 适配器类更改上传文件名为 newName.gif 的实现代码如下：

```
$adapter = new Zend_File_Transfer_Adapter_Http();                    //实例适配器类
$adapter->setDestination('./upfiles');                               //定义上传文件保存路径
$adapter->addFilter('Rename', array('target' => 'newName.gif', 'overwrite' => true), 'imagename');    //改文件名
$adapter->receive();                                                 //执行上传操作
```

25.6　Zend_Layout 网站布局

通过 Zend_Layout 网站布局，可以提高网站页面的重用率，便于网站内容的更新和维护。例如，在一个网站中，每个页面中都可能包含相同的头、尾文件，所以在每个页面中都需要编写相同的头、尾文件。如果应用 Zend_Layout 对页面进行布局，将页面的头、尾文件定义到单独的模板中，这样在创建页面时就不用再编写相同的头、尾文件代码，只要调用指定的头、尾模板文件即可。

25.6.1　Zend_Layout 概述

在 Zend Framework 中，使用 Zend_Layout 组件实现对网站视图层页面的布局，从而可以有效提高项目的可维护性、代码重用率和扩展能力，使用 Zend_Layout 对网站布局具有以下优点。

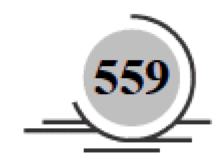
☑　建立 Zend Framework 的 MVC 框架后，可以自动配置 Zend_Layout 的解析。
☑　能够更好地实现业务逻辑和视图层的分离。
☑　Zend Framework 中允许在 application.ini 文件中配置布局名称、布局脚本及布局脚本路径。
☑　在不需使用布局的动作中，可以通过代码取消布局。
☑　Zend_Layout 在没有建立 Zend Framework 的 MVC 框架的前提下可以独立使用。

25.6.2　Zend_Layout 使用方法

以往在使用 PHP 开发 Web 应用时，实现网站的布局是使用文件包含函数将网站头部文件、尾部文件及其他模块包含到程序中。而使用 Zend_Layout 实现网站的布局，只需简单的配置即可。下面以 Zend Framework 的 MVC 框架为例，讲解 Zend_Layout 实现站点布局的方法和步骤。

（1）首先需要在 application.ini 文件中指定布局文件的保存目录和布局文件名。其代码如下：

```
resources.layout.layoutPath = APPLICATION_PATH "/layouts"
```



```
resources.layout.layout = "default"
```

上述配置代码指定布局文件的保存目录为应用路径下的 layouts 目录，并指定该目录下的 default.phtml 文件为默认布局文件。这里需要注意，在配置网站布局时，可以不指定布局文件，但需要在控制器的动作方法中通过如下代码指定。

```
$this->_helper->layout->setLayout('default');
```

（2）完成 application.ini 的布局配置后，就可以建立网站的头部导航（header.phtml）和尾部导航（footer.phtml）视图文件，为了能够通过 Zend_View 对象的 render()方法获取到这两个文件，需要将 header.phtml 和 footer.phtml 文件直接保存到框架的 scripts 目录下。

（3）完成网站头部导航和尾部导航建立后，就可以建立布局文件 default.phtml，根据步骤（1）中的配置，default.phtml 文件被保存在应用路径的 layouts 目录下。其关键代码如下：

```
<html>
    <head>
        <!--指定网站头信息-->
    </head>
    <body
        <!--包含头部导航-->
        <?php echo $this->render('default_header.phtml')?>
        <!--包含页面主体部分-->
        <?php echo $this->layout()->content;?>
        <!--包含尾部导航-->
        <?php echo $this->render('default_footer.phtml')?>
    </body>
</html>
```

从上述代码可知，在布局文件中包含网站的头部文件和尾部文件是通过视图对象的 render()方法实现的，包含网站的主体内容是通过视图对象的 layout()方法返回值的 content 属性实现的。

25.6.3 Zend_Layout 应用实例

为了能够更加清楚地掌握 Zend_Layout 组件实现网站布局的配置方法，下面通过具体实例进行讲解。

例 25.6 使用 Zend_Layout 实现对明日 IT 新闻网的页面布局。**（实例位置：光盘\TM\Instance\25\25.6）**

明日 IT 新闻网是一个专业的 IT 新闻类网站，在网站的首页、新闻列表页和新闻详细信息页面都使用相同的网站头部和尾部导航，为了提高代码的重用率及设计统一的页面风格，在实施网站布局时采用 Zend_Layout 组件实现。

（1）建立 Zend Framework 的 MVC 框架结构，具体创建过程请参见本章第 25.2 节的内容，这里不再赘述。

（2）打开 application.ini 文件，加入如下配置代码来指定布局文件的保存目录。

```
resources.layout.layoutPath = APPLICATION_PATH "/layouts"
```

（3）建立明日 IT 新闻网的首部导航视图和尾部导航视图，其效果分别如图 25.12 和图 25.13 所示，并分别将这两个视图文件命名为 header.phtml 和 footer.phtml。



图 25.12　网站头部导航



图 25.13　网站尾部导航

（4）在网站布局目录下建立 default.phtml 视图文件。其代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/
xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <?php
            echo $this->headMeta()->setHttpEquiv('content-type', 'text/html; charset=utf-8')
                                ->appendHttpEquiv('x-ua-compatible', 'ie=7');
        ?>
        <?php echo $this->headLink()->setStylesheet($this->baseUrl('/css/style.css'))?>
        <style type="text/css">
            .default-body                {background:url(<?php echo $this->baseUrl('/img/bg.jpg')?>)
repeat-x 0 0; background-color:#E7F6FB;}
            .default-top             {width:980px; height:30px; clear:both;}
            .default-top .li1                {width:600px; height:30px; line-height:30px; text-align:left;
color:#FFFFFF; float:left;}
            .default-top .li2              {width:200px; height:30px; line-height:30px; text-align:right;
color:#FFFFFF; float:right;}
            .default-main             {width:980px; background-color:#FFFFFF; clear:both;}
        </style>
    </head>
    <body class="default-body">
        <div class="default-top">
            <ul>
             </ul>
        </div>
        <div class="default-main">
            <?php echo $this->render('default_header.phtml')?>
            <?php echo $this->layout()->content;?>
            <?php echo $this->render('default_footer.phtml')?>
        </div>
        <div class="cell_h"></div>
    </body>
</html>
```

（5）建立 IndexController 控制器，并在其中建立 indexAction 动作。其代码如下：

```
<?php
class IndexController extends Zend_Controller_Action{
    public function indexAction(){
         $this->_helper->layout->setLayout('default');            //指定布局文件为 default.phtml
     }
}
```


通过上面步骤即可实现通过 Zend_Layout 组件对网站进行布局，本实例的运行效果如图 25.14 所示。



图 25.14　明日 IT 新闻网的网站布局效果

25.7 Zend_Paginator 分页

在制作网站的数据展示模块时，如果数据量较大，一般采用分页显示数据的方式，Zend_Framework 框架提供的 Zend_Paginator 组件是专门用于数据分页显示的，本节将对该组件进行介绍。

25.7.1 Zend_Paginator 简介

Zend_Paginator 是一个非常优秀的数据分页组件，该组件采用适配器方式，根据不同的适配器可以实现对数组、Zend_Select 查询结果的分页展示。该组件具有如下优点。

☑ 对任意数据进行分页，而不是针对关系型数据库。
☑ 针对需求数据进行呈现，而不是全部数据。
☑ 不强制用户只使用一种途径呈现数据和渲染分页控件。
☑ 可以单独使用（相对 Zend Framework 框架本身）。

25.7.2 Zend_Paginator 分页方法

使用 Zend_Paginator 组件实现数据的分页显示，可以简化操作，有效地提高程序的开发效率和后期维护工作。下面讲解使用 Zend_Paginator 实现数据分页的方法。

（1）使用 Zend_Paginator 组件实现数据分页，首先需要确定要使用的分页适配器，常用的分页适配器有数组适配器和 Zend_Select 分页适配器，其中数组适配器使用 Zend_Paginator_Adapter_Array 类指定，Zend_Select 查询结果的分页显示所需的适配器使用 Zend_Paginator_Adapter_DbSelect 指定。例如，使用 Zend_Paginator_Adapter_Array 适配器的代码如下：

```
$array=array();                                        //定义要分页的数组
$adapter = new Zend_Paginator_Adapter_Array($array);   //实例化适配器
```

（2）创建完分页适配器后，就对 Zend_Paginator 分页类进行实例化并指定分页参数。该过程的代码如下：

```
$paginator = new Zend_Paginator($adapter);             //实例化 Zend_Paginator 对象
$paginator->setPageRange(5)                            //指定分页控制器面板显示的页码个数
          ->setItemCountPerPage(20)                    //指定每页显示的记录数
          ->setCurrentPageNumber(1);                   //指定当前显示的页码
$this->view->paginator = $paginator;                   //将分页类赋值给视图变量
```

（3）获得 Zend_Paginator 分页类实例化对象后，就可以在视图层通过传递的分页类对象显示分页数据和分页控制面板，分页数据通过 foreach 语句循环输出。例如，通过如下代码显示分页数据中 ID 的值。

```
foreach($this->paginator as $data)                     //遍历分页类对象
{
    echo $date['id'];                                  //输出 ID 的值
}
```

（4）通过 foreach 语句循环输出指定页码的分页数据后，还需要制作用于实现分页控制的分页链接面板。分页链接控制面板文件需要放置在框架的 scripts 目录中，该文件主要用于生成分页控制面板视图，常用的页面属性如表 25.4 所示。

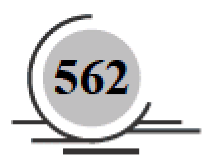


表 25.4　分页链接控制面板文件的参数说明

参　　数	说　　明
first	首页页码
firstItemNumber	当前页码中第一条数据在完整数据集中的序号（位置）
firstPageInRange	程序运行开始时实现的页码
current	当前页码
currentItemCount	当前页数据数
itmCountPerPage	本页最多数据数量
last	最后页码
lastItemNumber	当前页码中最后一条数据在完整数据集中序号（位置）
next	下一页页码
pageCount	总页数
pageInRange	现实页码数组
previous	上一页页码
totalItemCount	总数据数量

根据表 25.4 可知，在分页链接控制面板文件中可以通过如下代码显示“最后一页”导航链接。

```
<a href="<?php echo $this->baseUrl('/news/list?page='.$this->last)?>">最后一页</a>
```

（5）建立完分页链接控制面板页后，还需要在分页视图中使用视图对象的 paginationControl()方法导入分页面板控制页面。实现导入分页面板页面的代码如下：

```
<?php echo $this->paginationControl($this->paginator, 'Sliding', 'list_news_pagination_control.phtml',
array('flag'=>$this->flag, 'order'=>$this->order))?>
```

paginationControl()方法一般有 4 个参数，其中第 1 个参数用于指定分页控制面板关联的 Zend_Pageinator 对象；第 2 个参数用于指定控制面板样式，其中常用的样式及说明如表 25.5 所示。第 3 个参数用来指定分页控制面板的页面文件名称；最后一个参数为数组型，用于向分页面板文件传递视图变量。

表 25.5　分页样式详细说明

参　　数	说　　明
All	返回每个页码，这是下拉菜单跳转中的分页
Elastic	类似于 Google 分页样式
Jumping	跳转到最后一页后，返回到首页页码
Sliding	类似于 Yahoo 分页样式，以当前页为中心（推荐）

25.7.3　Zend_Paginator 分页应用

通过上节的学习可以掌握使用 Zend_Paginator 实现数据分页的基本思路和方法，本节将通过具体实例讲解如何使用 Zend_Paginator 实现分页。

例 25.7　使用 Zend_Paginator 实现对明日 IT 新闻网的新闻主题的分页显示。**（实例位置：光盘\TM\25\25.7）**

在制作明日 IT 新闻网的新闻主题展示页面时，使用 Zend_Paginator 组件实现主题的分页显示，下面具体讲解该模块的制作方法。

（1）建立 Zend Framework 的 MVC 框架结构，具体创建过程请参见本章第 25.2 节的内容，这里不再赘述。

（2）建立新闻表模型 Model_DbTable_News 类，并在该类中建立用于实现新闻分页的 findByPage()方法。application\models\DbTable News.php 文件的代码如下：

```
public function findByPage ($flag, $orderIndex, $page = 1, $itemCountPerPage = 10, $pageRange = 5)
{
    $select = $this->getAdapter()->select();                          //生成 select 对象
    $select->from($this->_name);                                      //指定要查询的表名
    if ($flag !== 'all') {                                            //指定查询条件
        if ($flag === 'ttj') {
        $where = 'istodaycommend = true';
        } else {
        $where = $this->getAdapter()->quoteInto('flag = ?', $flag);
        }
        $select->where($where);
    }
    if ($orderIndex == 0) {                                           //排序方式
        $order = 'addtime desc';
    } elseif ($orderIndex == 1) {
        $order = 'addtime';
    } elseif ($orderIndex == 2) {
        $order = 'browse desc';
    } elseif ($orderIndex == 3) {
        $order = 'browse';
    }
    $select->order($order);                                           //排列顺序
    $paginatorAdapter = new Zend_Paginator_Adapter_DbSelect($select);  //实例并构建 Zend_Paginator 对象
    $paginator = new Zend_Paginator($paginatorAdapter);
$paginator->setPageRange($pageRange)->setItemCountPerPage($itemCountPerPage)->setCurrentPageNum
ber($page);                                                           //指定分页参数
    return $paginator;                                                //返回 Zend_Paginator 对象
}
```

上述代码首先构建 Zend_Select 查询对象，然后使用 Zend_Paginator_Adapter_DbSelect 分页适配器生成用于对 Zend_Select 查询结果进行分页的适配器对象，同时实例化 Zend_Paginator 分页类对象并指定分页参数，最后返回分页类对象。

（3）在 IndexController 控制器的 indexAction 动作中，调用新闻表模型的 findByPage()方法，并将返回结果赋值给视图变量，以便在视图层应用。application\controllers\IndexController.php 文件的代码如下：

```
public function indexAction ()
{
    //接收参数
    if($this->_request->getParam('flag')==""   ||   $this->_request->getParam('order')==""   ||$this->_request->
getParam('page')==""){
        $flag ="all";
        $order = 2;
        $page = 1;
    }else{
        $flag = $this->_request->getParam('flag');
        $order = $this->_request->getParam('order');
        $page = $this->_request->getParam('page');
    }
    //分页参数
    $itemCountPerPage = 10;
    $pageRange = 5;
    //视图变量赋值
    $this->view->flag = $flag;
    $this->view->order = $order;
    //新闻分页
    $paginator = $this->_news->findByPage($flag, $order, $page, $itemCountPerPage, $pageRange);
```

```
        $this->view->paginator = $paginator;
        $this->_helper->layout->setLayout("default");
}
```

（4）在视图层，使用 foreach 循环语句输出新闻主题信息。application\views\scripts\index\index.phtml 文件的代码如下：

```
<?php foreach ($this->paginator as $news):?>
    <div class="content">
        <div class="title">
            ·<a href="#" class="a25"><?php echo $this->escape($news['title'])?></a>
        </div>
        <div class="description">
            <?php echo trim($this->escape($this->substr($news['uncontent'], 220)))?> <a href="#" class="a3">
<nobr>&lt;详细&gt;</nobr></a>
        </div>
        <div class="msg">
            来源：<font color="#888888"><?php echo $this->escape($news['source'])?></font> | 作者：<font
color="#888888"><?php echo $this->escape($news['author'])?></font> | 时间：<font color="#888888"><?php
echo substr($news['addtime'], 0, 16)?></font> | 浏览：<font color="#888888"><?php echo $news['browse']?>
</font> 次
        </div>
    </div>
<?php endforeach;?>
```

（5）在 index.phtml 页面中使用 paginatorControl()方法调用分页控制面板。其代码如下：

```
<?php echo $this->paginationControl($this->paginator, 'Sliding', 'list_news_pagination_control.phtml',
array('flag'=>$this->flag, 'order'=>$this->order))?>
```

（6）建立分页控制面板。application\views\scripts\list_news_pagination_control.phtml 文件的代码如下：

```
<div style="width:280px; height:20px; line-height:20px; color:#0257AE; font-weight:bold; text-align:center;
border:1px solid #92C7ED; background-color:#FAFEFF; margin-right:8px; float:left;">
每页 <font style="color:#999999; font-family:Arial; font-size:12px;"><?php echo $this->itemCountPerPage?>
</font>  条 / 共  <font style="color:#999999; font-family:Arial; font-size:12px;"><?php echo
$this->totalItemCount?></font> 条 第 <font style="color:#999999; font-family:Arial; font-size:12px;">
<?php echo $this->current?></font>  页 / 共  <font style="color:#999999; font-family:Arial;
font-size:12px;"><?php echo $this->pageCount?></font> 页
</div>
<?php if($this->totalItemCount > 0):?>
    <?php if($this->current > 1):?>
    <div style="width:40px; height:20px; line-height:20px; border:1px solid #92C7ED; background-color:<?php
if($this->current==1):?>#1A75D2;<?php else:?>#FAFEFF;<?php endif;?> font-weight:bold; float:left; margin-
right:8px;">
        <a href="<?php echo $this->baseUrl('/list-'.$this->flag.'-'.$this->order.'-1.html')?>" class="<?php
if($this->current==1):?>a4<?php else:?>a3<?php endif;?>">首页</a>
    </div>
    <?php endif;?>
    <?php foreach ($this->pagesInRange as $page):?>
    <div style="width:20px; height:20px; line-height:20px; border:1px solid #92C7ED; background-color:<?php
if($this->current==$page):?>#1A75D2;<?php else:?>#FAFEFF;<?php endif;?> font-family:Arial; font-style:italic;
font-weight:bold; float:left; margin-right:8px;">
        <a href="<?php echo $this->baseUrl('/list-'.$this->flag.'-'.$this->order.'-'.$page.'.html')?>" class="<?php
if($this->current==$page):?>a4<?php else:?>a3<?php endif;?>"><?php echo $page?></a>
    </div>
    <?php endforeach;?>
    <?php if($this->pageCount > $this->pageRange && $this->pageCount!=$this->current):?>
    <div style="width:40px; height:20px; line-height:20px; border:1px solid #92C7ED; background-color:<?php
if($this->current==$this->last):?>#1A75D2;<?php else:?>#FAFEFF;<?php endif;?> font-weight:bold; float:left;
```

```
margin-right:8px;">
            <a href="<?php echo $this->baseUrl('/list-'.$this->flag.'-'.$this->order.'-'.$this->last.'.html')?>" class="<?php if
($this->current==$this->last):?>a4<?php else:?>a3<?php endif;?>">尾页</a>
      </div>
      <?php endif;?>
<?php endif;?>
```

以上为明日 IT 新闻网新闻主题循环输出的实现过程，运行本实例，效果如图 25.15 所示。



图 25.15　明日 IT 新闻网新闻主题分页展示

25.8　实　　战

视频讲解：光盘\TM\Video\第 25 章\实战.exe

25.8.1　Zend_Paginator 实现数据分页显示

例 25.8　在本实例中，应用 Zend_Paginator 对数据库中数据进行分页显示，其运行结果如图 25.16 所示。**（实例位置：光盘\TM\Instance\25\25.8）**



ID	学号	姓名	年龄	班级
1	20090801	Liuning	27	2.6班
2	20090802	Pankaihua	28	2.6班
3	20090803	Lisan	27	2.6班
4	20090804	Liuzhonghua	28	2.6班
5	20090803	wangermazi	27	2.6班

图 25.16　Zend_Paginaator 分页显示

具体步骤如下。

（1）搭建 Zend Framework MVC 环境。

（2）Zend_Config 配置站点初始参数。



（3）在 application\controllers\IndexController.php 中加入如下代码：

```
  $pageNumber = 5;
❶$paginator = Zend_Paginator::factory($this->view->studentsModel);
  $paginator->setItemCountPerPage($pageNumber);
  $paginator->setCurrentPageNumber($this->_getParam('page'));
  $paginator->setPageRange(5);
❷Zend_Paginator::setDefaultScrollingStyle('Sliding');
  $paginator->setView($this->view);
  $this->view->studentsModel = $paginator;
❸$this->view->paginator = $paginator;
  $this->render();
```

说明

❶ Zend_Paginator::factory()为分页器的工厂方法，参数为数组中读取出的数据（数组）。

❷ Zend_Paginator::setDefaultScrollingStyle('Sliding')调用页码样式。

❸ $this->view->paginator 将分页传给 V 层。

（4）在 application\views\scripts\index 文件下创建 pagelist.phtml 文件，建立所需要的分页页码。其代码如下：

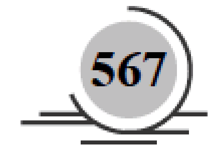
```
<?php if ($this->pageCount): ?>
<div class="paginationControl">
<!-- Previous page link -->
<?php if (isset($this->previous)): ?>
   <a href="<?php echo $this->url(array('controller'=>'index', 'action'=>'index', 'page'=>$this->previous)); ?>"><
上一页</a> |
<?php else: ?>
   <span class="disabled">< 上一页</span> |
<?php endif; ?>
<!-- Numbered page links -->
<?php foreach ($this->pagesInRange as $page): ?>
  <?php if ($page != $this->current): ?>
     <a href="<?php echo $this->url(array('controller'=>'index', 'action'=>'index', 'page'=>$page)); ?>"><?=
$page; ?></a> |
  <?php else: ?>
    <?= $page; ?> |
  <?php endif; ?>
<?php endforeach; ?>
<!-- Next page link -->
<?php if (isset($this->next)): ?>
   <a href="<?php echo $this->url(array('controller'=>'index', 'action'=>'index', 'page'=>$this->next)); ?>">下一
页 ></a>
<?php else: ?>
   <span class="disabled">下一页  ></span>
<?php endif; ?>
</div>
<?php endif; ?>
```

25.8.2　Zend Framework 用户身份验证

例 25.9　使用 Zend Framework 框架实现用户身份的验证主要应用 Zend_Auth 组件。明日 IT 新闻网的用户登录模块就是应用 Zend_Auth 组件对用户身份进行验证。**（实例位置：光盘\TM\Instance\25\25.9）**

其具体步骤如下。

（1）建立 Zend Framework 的 MVC 框架结构。



（2）建立用户登录表单。该表单中元素包括用户昵称文本框、用户密码文本框和验证码文本框，实现该过程的代码主要由 HTML 语句的表单相关标签组成，具体实现代码请详见本书附带光盘，其运行效果如图 25.17 所示。



图 25.17　明日 IT 新闻网的登录页面

（3）在用户表模型中建立 isValid()方法，用于验证登录信息是否正确，正确则返回 true，反之返回 false。application\models\DbTable\News.php 文件的关键代码如下：

```
public function isValid ($netname, $password)
{
    //构建验证适配器
    $authAdapter = new Zend_Auth_Adapter_DbTable($this->getAdapter());
$authAdapter->setTableName($this->_name)->setIdentityColumn('netname')->setCredentialColumn('password'
)->setIdentity($netname)->setCredential($password);
    //获取 Zend_Auth 对象并验证
    $auth = Zend_Registry::get('auth');
    return $auth->authenticate($authAdapter)->isValid();
}
```

（4）当用户在用户登录表单中输入用户信息并单击“登录”按钮后，登录信息将被提交到 UserController 控制器的 loginAction 动作中验证用户的登录信息是否正确。application\modules\default\controllers\UserController.php 文件的代码如下：

```
if ($this->_request->isPost()) {
    //接收用户名和密码
    $netname = trim($this->_request->getParam('netname'));
    $password = trim($this->_request->getParam('password'));
    //对用户身份进行验证
    if ($this->_user->isValid($netname, md5($password))) {
        $url = $this->_request->getParam('url');
        if ($url == null) {
            //定位到登录成功页面
            $this->_redirect('/user/success/flag/login');
        } else {
            $url = str_replace('@', '?', str_replace('$', '&', $url));
            $this->_redirect($url);
        }
        exit();
    } else {
        $errMsg = '登录昵称或密码输入有误，请重新登录！';
        $this->view->errMsg = $errMsg;
```

```
            $this->view->netname = $netname;
        }
}
```

25.8.3　Zend_Mail 发送邮件

例 25.10　Zend Framework 通过 Zend_Mail 组件发送邮件，邮件在 Zend_Mail 里通过默认的 Zend_Mail_Transport_Sendmail 和 Zend_Mail_Transport_Smtp 来发送。在本实例中通过 Zend_Mail_Transport_Smtp 来发送邮件。**（实例位置：光盘\TM\Instance\25\25.10）**

发送邮件应用 Zend_Mail_Transport_Smtp($ip)，$ip 地址为服务器 IP 地址。Zend_Mail 发送邮件有 4 个基本属性，如表 25.6 所示。

表 25.6　Zend_Mail 发送邮件的 4 个基本属性

值	说　明
addTo($option,$to)	收件人地址，$option 为发件人地址，$to 为邮件中发件人地址
serFrom($option,$from)	发件人地址，$option 为收件人地址，$from 为邮件中收件人地址
setSubject($subject)	邮件标题
setbodyText($body)	邮件内容

其操作步骤如下。

（1）搭建 Zend Framework MVC 环境。

（2）Zend_Config 配置站点初始参数。

（3）在 application\models\DbTable 文件下创建 MailForm.php 文件，创建 4 个表单：收件人、发件人、标题和内容。其代码如下：

```
class Model_DbTable_MailForm extends Zend_Form
{
    public function __construct($option = null)
    {
        parent::__construct($option);
        $this->setName('form1');
        //form 控制器需要知道有哪些内容
        $to = new Zend_Form_Element_Text('to');
        $to->setLabel('收件人')
            ->setRequired(true)
            ->addFilter('StripTags')
            ->addFilter('StringTrim')
            ->addValidator('NotEmpty');
        $from = new Zend_Form_Element_Text('from');
        $from->setLabel('发件人')
            ->setRequired(true)
            ->addFilter('StripTags')
            ->addFilter('StringTrim')
            ->addValidator('NotEmpty');
        $subject = new Zend_Form_Element_Text('subject');
        $subject->setLabel('标题')
            ->setRequired(true)
            ->addFilter('StripTags')
            ->addFilter('StringTrim')
            ->addValidator('NotEmpty');
        $content = new Zend_Form_Element_Textarea('content');
        $content->setLabel('内容')
```

```
                ->setRequired(true)
                ->addFilter('StripTags')
                ->addFilter('StringTrim')
                ->addValidator('NotEmpty');
        $submit = new Zend_Form_Element_Submit('submit');
        $submit->setAttrib('id','submitbutton');
        $this->addElements(array($to,$from,$subject,$content,$submit));
    }
}
```

（4）在 application\controllers 文件下创建 IndexController.php，确定使用类及服务器 IP 地址，实例化 Zend_Mail，填入表单元素，发送邮件。其代码如下：

```
public function indexAction()
{
    $form = new Model_DbTable_MailForm();
    $form->submit->setLabel('发  送');
    $this->view->form = $form;
    if ($this->_request->isPost())
    {
        $formData = $this->_request->getPost();
        if ($form->isValid($formData))
        {
            $tr = new Zend_Mail_Transport_Smtp('192.168.1.247');
            $mail = new Zend_Mail();

            $mail->addTo('cym3100@163.com',$form->getValue('to'));
            $mail->setFrom('cym3100@163.com',$form->getValue('from'));
            $mail->setSubject($form->getValue('subject'));
            $mail->setBodyText($form->getValue('content'));
            $mail->send($tr);

            $this->_redirect('/');
        }else
        {
            $form->populate($formData);
        }
    }
}
```

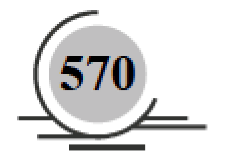
（5）在 application\views\scripts\index 文件下创建 index.phtml，在该页面中调用 Form 表单。其代码如下：

```
<?php echo $this->form; ?>
```

运行结果如图 25.18 所示。



图 25.18　Zend_Mail 发送邮件



说明 Zend_Mail 发送邮件需要实例化 Zend_Mail_Transport_Smtp($ip)，这个 IP 地址必须为服务器 IP。AddTo()和 AddFrom()中的第一个参数必须是服务器中的一个邮箱名称。

25.8.4　Zend_Form 制作用户注册表单

例 25.11　Zend_Form 是 Zend Framework 自带的组件，为 Web 程序中简化表单创建和处理，它可以将输入的元素过滤和校验，还可以对表单元素进行排序、解析。运行本实例将看到完整的 Zend_Form 用户注册表单。**（实例位置：光盘\TM\Instance\25\25.11）**

具体实现步骤如下。

（1）搭建 Zend Framework MVC 环境。

（2）Zend_Config 配置站点初始参数。

（3）在 application\models\DbTable 文件下创建 StudentForm.php 文件，建立 Zend_Form，确定数据库表名，设置表单元素。其代码如下：

```
class Model_DbTable_StudentForm extends Zend_Form
{
	public function __construct($option = null)
	{
		parent::__construct($option);
		$this->setName('tb_students');
		$id = new Zend_Form_Element_Hidden('id');
		$num = new Zend_Form_Element_Text('num');
		$num->setLabel('学号')
			->setRequired(true)
			->addFilter('StripTags')
			->addFilter('StringTrim')
			->addValidator('NotEmpty');
		$name = new Zend_Form_Element_Text('name');
		$name->setLabel('姓名')
			->setRequired(true)
			->addFilter('StripTags')
			->addFilter('StringTrim')
			->addValidator('NotEmpty');
		$age = new Zend_Form_Element_Text('age');
		$age->setLabel('年龄')
			->setRequired(true)
			->addFilter('StripTags')
			->addFilter('StringTrim')
			->addValidator('NotEmpty');
		$class = new Zend_Form_Element_Text('class');
		$class->setLabel('班级')
			->setRequired(true)
			->addFilter('StripTags')
			->addFilter('StringTrim')
			->addValidator('NotEmpty');
		$submit = new Zend_Form_Element_Submit('submit');
		$submit->setAttrib('id','submitbutton');
		$this->addElements(array($id,$num,$name,$age,$class,$submit));
	}
}
```

（4）在 application\controllers 文件下创建 IndexController.php，建立 Form，并以 POST 方式得到表单中元素，存入数据库中，返回本页面。其代码如下：

```
public function indexAction()
    {
        $form = new Model_DbTable_StudentForm();
        $form->submit->setLabel('注  册');
        $this->view->form = $form;
        if ($this->_request->isPost())
        {
            $formData = $this->_request->getPost();
            if ($form->isValid($formData))
            {
                $students = new Model_DbTable_Students();
                $row = $students->createRow();
                $row->num = $form->getValue('num');
                $row->name = $form->getValue('name');
                $row->age = $form->getValue('age');
                $row->class = $form->getValue('class');
                $row->save();
                $this->_redirect('/');
            }else
            {
                $form->populate($formData);
            }
        }
```

（5）在 application\views\scripts\index 下建立 index.phtml 文件，文件中只需要调用控制层中的显示 form 即可实现。代码如下：

```
<?php echo $this->form; ?>
```

注意 对于 Model 层中的文件，如果类的名称前加入 DbTable，则相应地在 models 文件下建立 DbTable 文件，将类文件建在下面；反之则不必建立。

运行结果如图 25.19 所示。



图 25.19　Zend_Form 用户注册表单



25.9　小　　结

本章主要介绍 Zend Framework 框架的 MVC 环境搭建、配置、框架结构、分页，并通过实例，对 Zend Framework 的各种应用进行了讲解，以此来增加读者对 Zend Framework 的理解。希望通过本章的学习，读者能够掌握 Zend Framework 技术，并将其灵活地运用到实际的网站开发中。

25.10　学习成果检验

1．Zend_Db_Table 如何执行添加、删除和修改操作？**（实例位置：光盘\TM\Instance\25\25.12）**

2．如何比较缓存日期与本地日期？**（实例位置：光盘\TM\Instance\25\25.13）**

第26章

综合实例（五）——电子商务网站

（视频讲解：53 分钟）

随着 PC 机（个人计算机）的发展和互联网的普及，电子商务从报文时代进入到了 Internet 时代，并逐渐被大众所了解和接受。电子商务（Electronic Commerce，简称 EC），是目前发展较快的一种商务模式。迄今为止，不同领域的人对 EC 的理解各有不同。简单地说，EC 是一种基于 Internet，利用计算机硬件、软件等现有设备和协议进行各种商务活动的方式。

通过阅读本章内容，你可以：

- 了解如何分析并设计数据库
- 了解 Ajax 无刷新验证技术
- 了解使用 JS 脚本获取、输出标签内容
- 熟悉在新窗口使用 session
- 熟悉 smarty 模板配置类文件
- 熟悉分页技术
- 熟悉购物车模块概述

26.1　电子商务网站概述

20 世纪 90 年代，互联网的蓬勃发展为企业提供了一个全新的机遇。企业网站、电子商务成为热门话题。其中，电子商务更是关系到经济结构、产业升级和国家整体经济竞争力。为此，我国已经将发展电子商务列为信息化建设的重要内容并努力创造条件，积极地推进电子商务的发展。

据美国在线（AOL）和 Henley Centre 联合进行的一项调查显示：国外有 80%的受调查者会选择网上购物或寻求帮助，10%的受调查者会选择熟悉的品牌或厂商来购买。而在国内，自 1997 年拉开了电子商务的序幕，短短的 10 年里，全国已有 4 万家商业网站，几乎每天都有新的网站诞生，厂商所在地也从上海、广州、深圳等沿海发达地区扩展到全国各地区。

26.2　系统分析

视频讲解：光盘\TM\Video\第 26 章\系统分析.exe

26.2.1　系统目标

根据客户提供的需求和对实际情况的考察与分析，该电子商务应该具备如下特点。

- ☑ 首页设计要能够吸引用户的目光，整个页面要以简洁为主，突出重点。
- ☑ 可操作性强，避免复杂的、有异议的链接。
- ☑ 浏览速度快，尽量避免长时间打不开页面的情况发生。
- ☑ 商品信息部分有实物图例，图像清楚、文字醒目。
- ☑ 详细的商品查询功能，可以通过商品的各个属性来搜索。
- ☑ 详细的流程介绍，从浏览商品到购买结账，各个步骤之间的联系最好能以图例来说明。
- ☑ 提供在线咨询。
- ☑ 后台可以对用户信息和商品信息进行详尽的查看和管理。
- ☑ 订单管理。
- ☑ 易维护，并提供二次开发支持。

26.2.2　功能流程结构

电子商务平台分前台系统和后台系统。下面分别给出前、后台的系统功能结构图。电子商务前台系统功能结构如图 26.1 所示。

电子商务后台系统功能结构图如图 26.2 所示。

26.2.3　程序预览

电子商务网站由多个功能模块组成，为了让读者对本系统有个初步的了解和认识，下面列出几个典型功能的页面，其他页面参见光盘中的源程序。

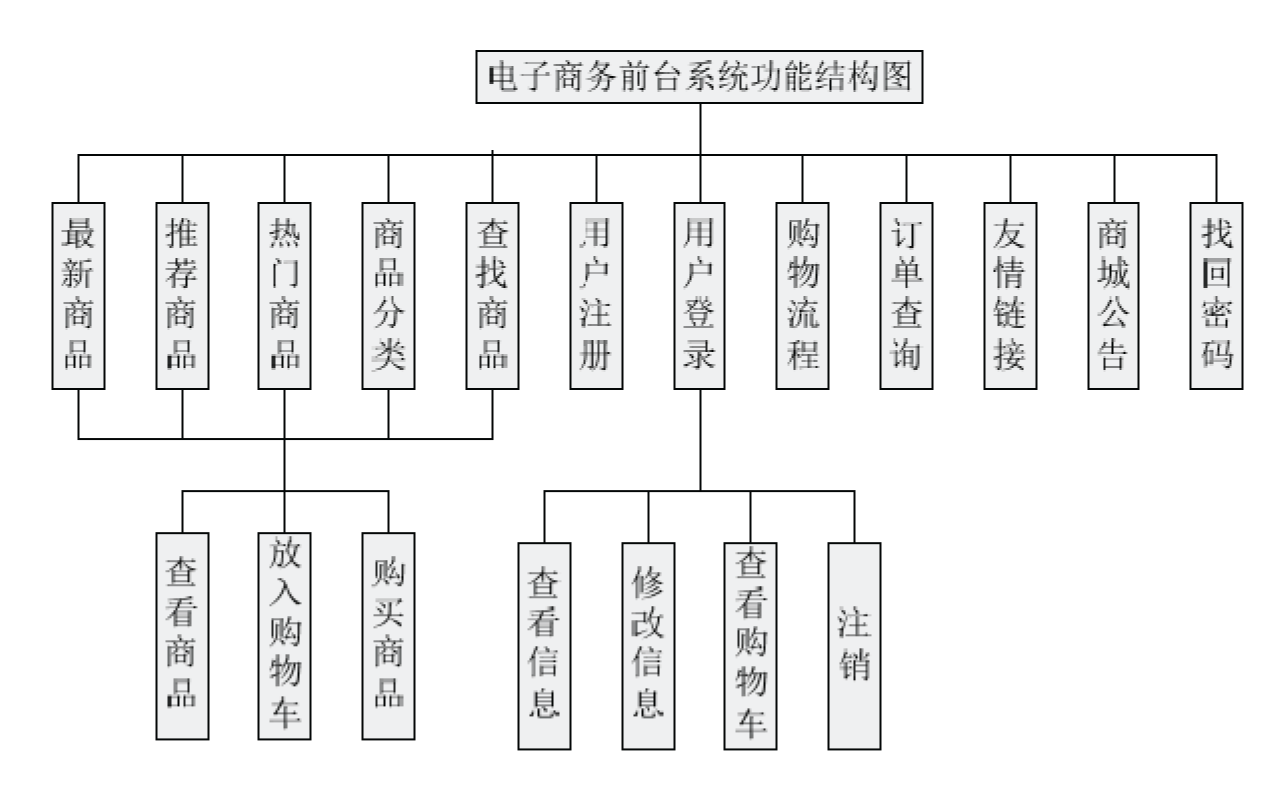


图 26.1　电子商务前台系统功能结构图

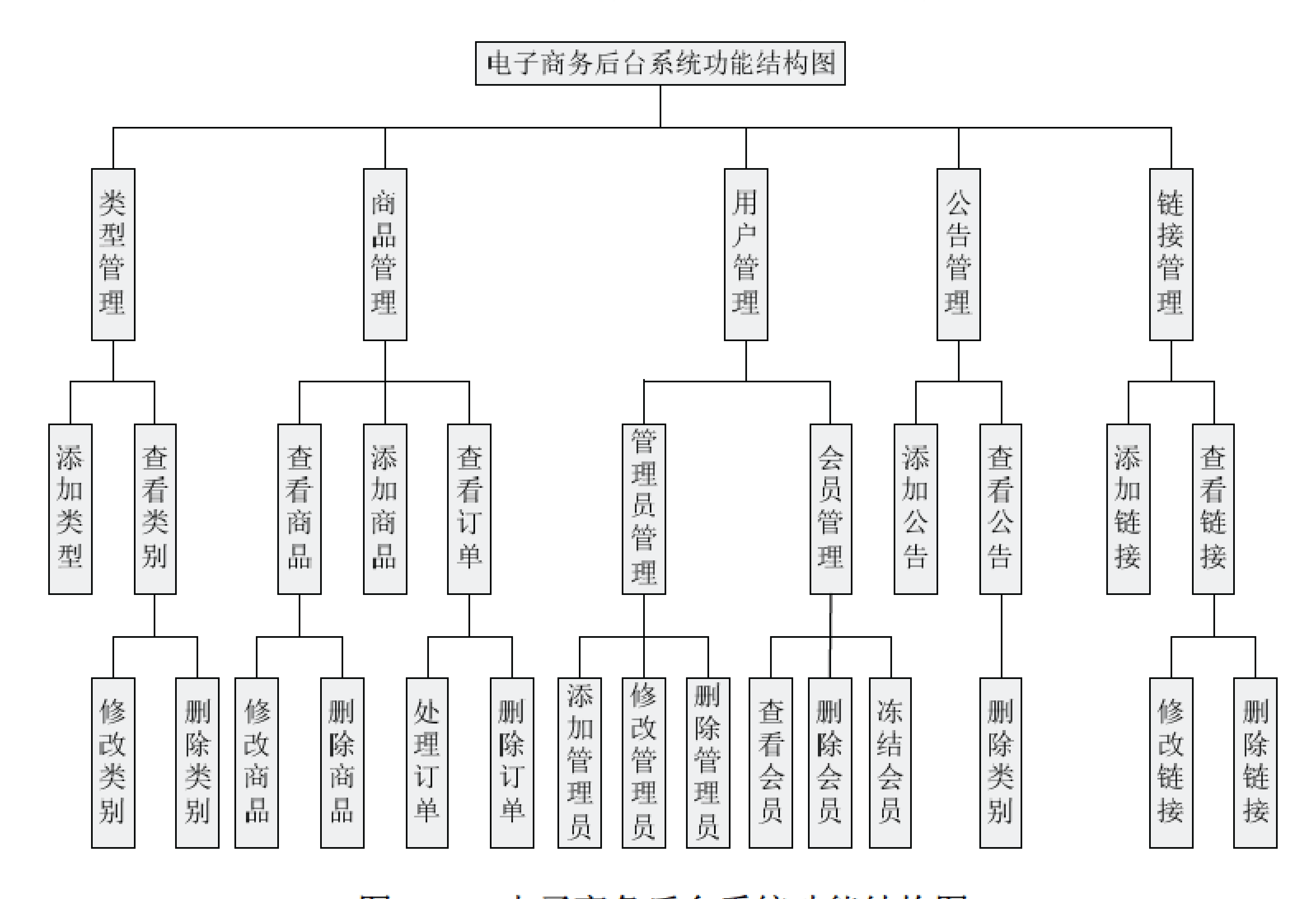


图 26.2　电子商务后台系统功能结构图

明日购物商城系统主页如图 26.3 所示，展示网站的部分最新商品、热门商品、推荐商品以及网站的最新公告和会员登录窗口。



图 26.3　明日购物商城首页

推荐商品展示页面如图 26.4 所示，分页展示网站的所有推荐商品。

购物车页面如图 26.5 所示，展示会员在本站购买的商品。



图 26.4　推荐商品展示

我的购物车

	商品名称	购买数量	市场价格	会员价格	折扣率	合计
□	自行车	3	388	349.2	9	1047.6
□	数码相机	5	1888	1699.2	9	8496
□	洗衣机	1	2666	2399.4	9	2399.4
□	家庭影院	1	4888	4399.2	9	4399.2

全选　反选　删除选择　继续购物　去收银台　共计：16342.2 元

图 26.5　购物车

用户注册页面如图 26.6 所示，完成用户注册信息的填写和验证操作。

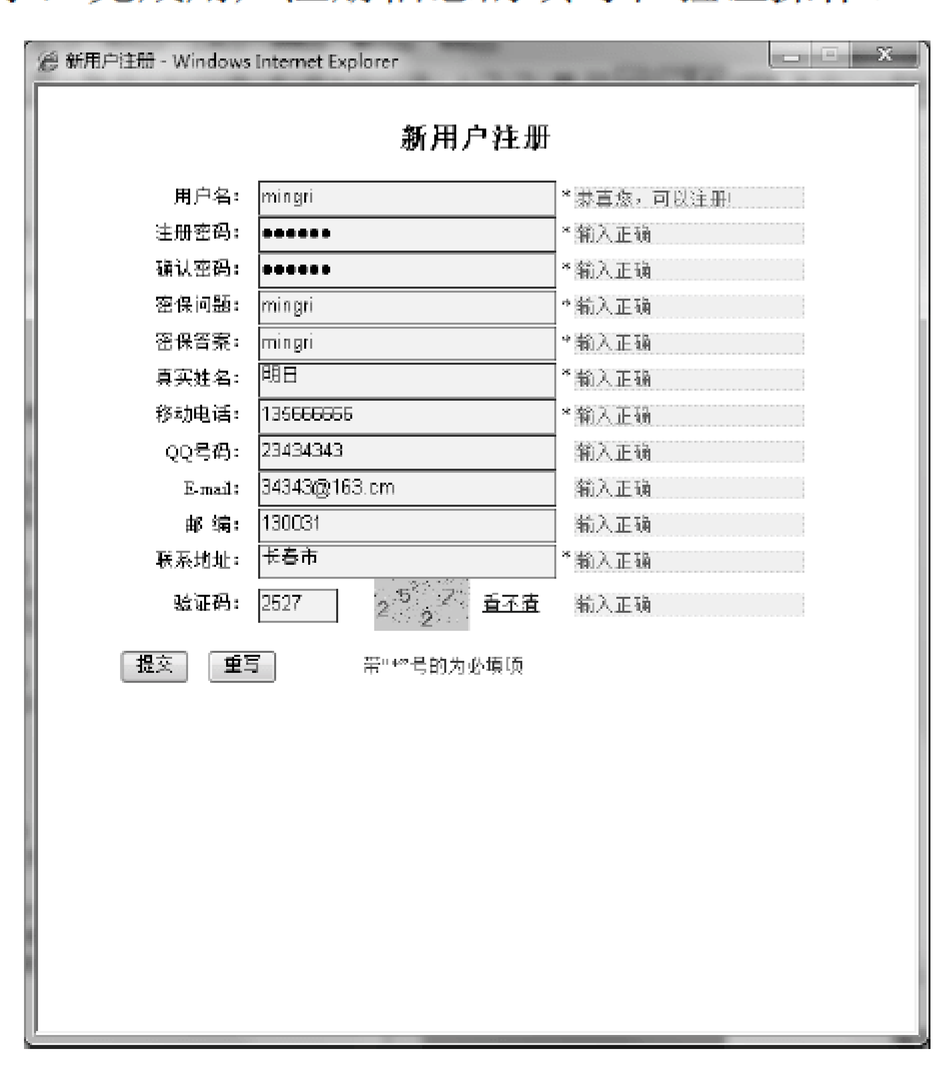


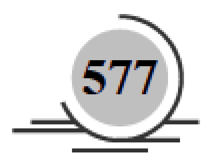
图 26.6　用户注册页面

26.3　数据库设计

无论是什么系统软件，其最根本的功能就是对数据的操作与使用。所以，一定要先做好数据的分析、设计与实现，然后才实现对应的功能模块。

26.3.1　数据库分析

根据需求分析和系统的功能流程图，找出需要保存的信息数据（也可以理解为现实世界中的实体），



并将其转化为原始数据（属性类型）形式。这种描述现实世界的概念模型，可以使用 E-R 图来表示，也就是实体-联系图。最后将 E-R 图转换为关系数据库。这里重点介绍几个 E-R 图。

1. 会员信息实体

会员信息实体包括编号、用户名、密码、E-mail、身份证号、联系电话、QQ 号、密码保护、密码答案、邮编、注册时间、真实姓名等属性。会员信息实体 E-R 图如图 26.7 所示。

2. 商品信息实体

商品信息实体包括编号、名称、上市时间、添加日期、型号、图片、库存、销售量、商品类型、会员价、市场价、是否打折等属性。商品信息实体 E-R 图如图 26.8 所示。

除上面介绍的两个 E-R 图，还有商品订单实体、商品评价实体、公告实体、管理员实体和类型实体和友情链接实体等，限于篇幅，这里仅列出主要的实体 E-R 图。

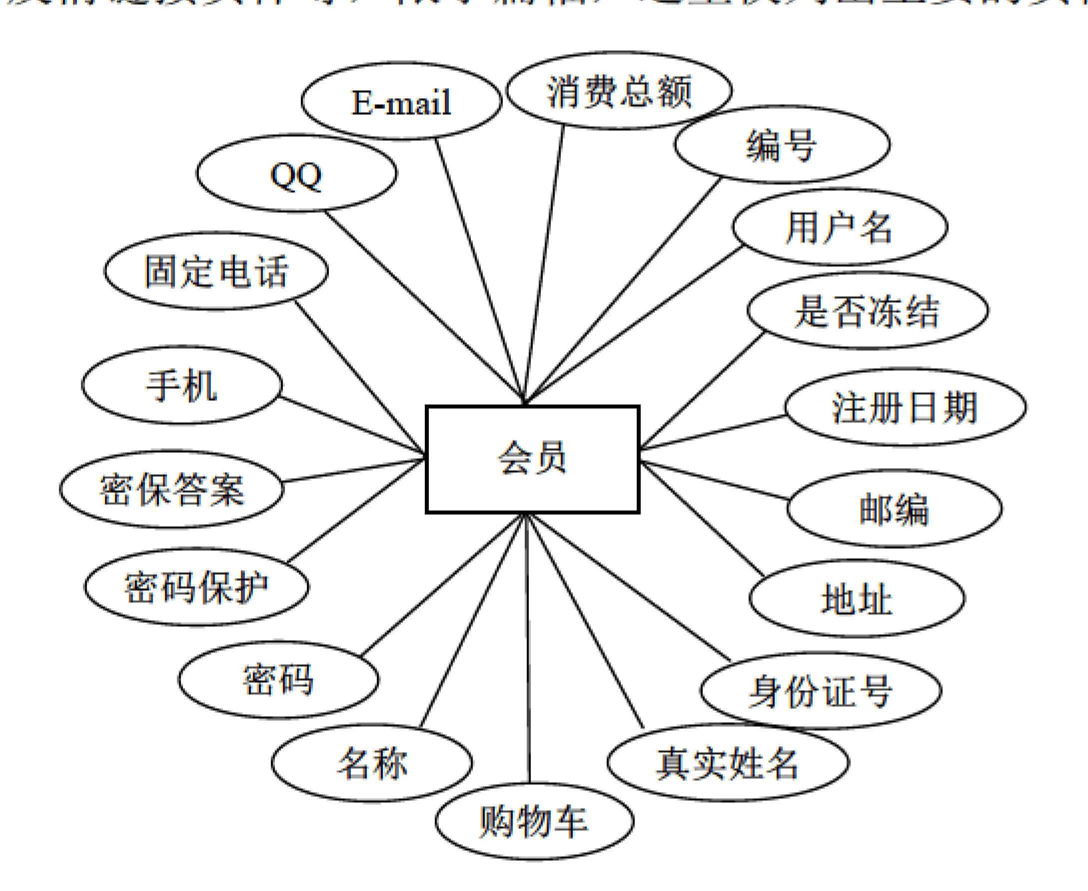


图 26.7 会员信息实体 E-R 图

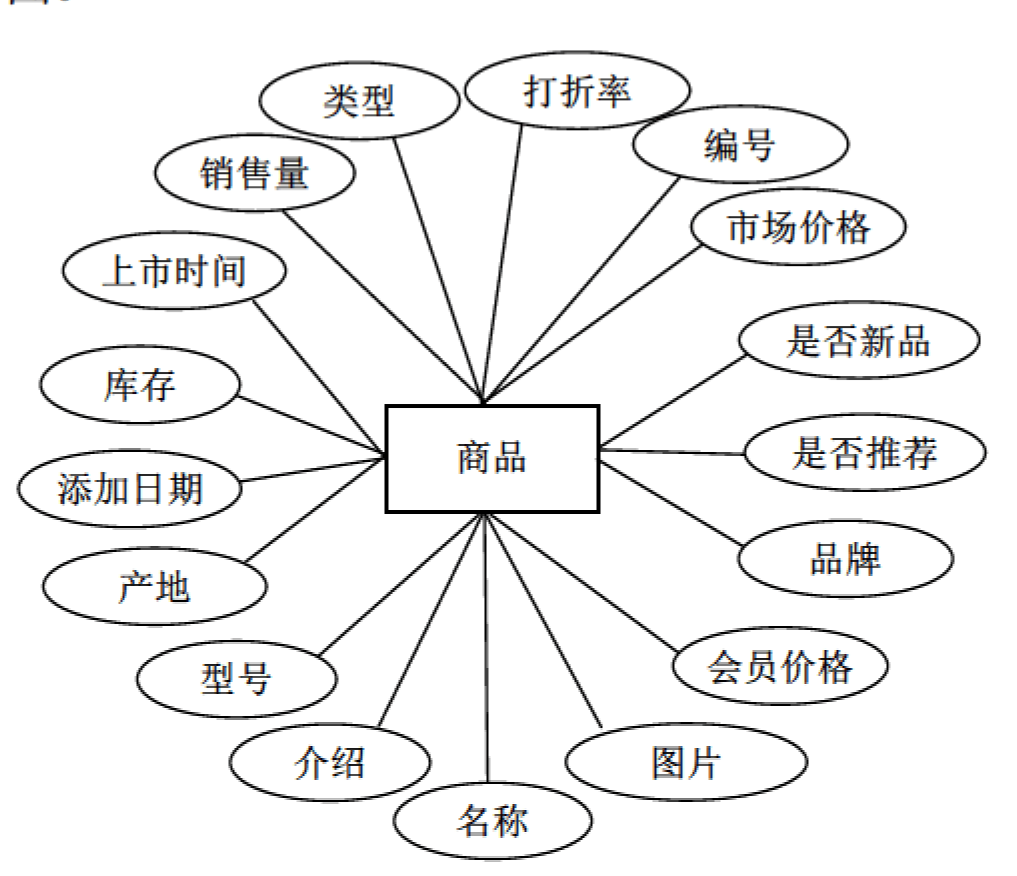


图 26.8 商品信息实体 E-R 图

26.3.2 创建数据库和数据表

视频讲解：光盘\TM\Video\第 26 章\创建数据库和数据表.exe

系统 E-R 图设计完成后，接下来根据 E-R 图来创建数据库和数据表。首先来看一下电子商务平台所使用的数据表情况，如图 26.9 所示。

下面来看各个数据表的结构和字段说明。

☑ tb_admin（管理员信息表）

管理员信息表主要用于存储管理员的信息，其结构如图 26.10 所示。

表	操作						记录数	类型	整理
tb_admin							1	MyISAM	utf8_unicode_ci
tb_class							6	MyISAM	utf8_unicode_ci
tb_commo							6	MyISAM	utf8_unicode_ci
tb_form							14	MyISAM	utf8_unicode_ci
tb_links							3	MyISAM	utf8_unicode_ci
tb_opinion							0	MyISAM	utf8_unicode_ci
tb_public							5	MyISAM	utf8_unicode_ci
tb_user							3	MyISAM	utf8_unicode_ci

图 26.9 电子商务数据表



字段	类型	整理	属性	Null	默认	额外
id	int(4)			否		auto_increment
name	varchar(50)	utf8_unicode_ci		否		
pwd	varchar(50)	utf8_unicode_ci		否		

图 26.10 管理员信息表结构

☑ tb_class（商品类型列表）

商品类型列表主要用于添加商品的类别，可以设定多个子类别（目前最多只能到二级子类别），其结



构如图 26.11 所示。

☑　tb_commo（商品信息表）

商品信息表主要用于存储关于商品的相关信息，其结构如图 26.12 所示。

字段	类型	整理	属性	Null	默认	额外
id	int(4)			否		auto_increment
name	varchar(20)	utf8_unicode_ci		否		
supid	int(2)			否		

图 26.11　商品类型列表结构

字段	类型	整理	属性	Null	默认	额外
id	int(4)			否		auto_increment
name	varchar(50)	utf8_unicode_ci		否		
pics	varchar(200)	utf8_unicode_ci		否	pics/null.jpg	
info	mediumtext	utf8_unicode_ci		否		
addtime	date			否		
area	varchar(50)	utf8_unicode_ci		否		
model	varchar(50)	utf8_unicode_ci		否		
class	varchar(50)	utf8_unicode_ci		否		
brand	varchar(50)	utf8_unicode_ci		否		
stocks	int(4)			否	1	
sell	int(4)			否	0	
m_price	float			否		
v_price	float			否		
fold	float			否	9	
isnew	int(1)			否	1	
isnom	int(1)			否	0	

图 26.12　商品信息表结构

此外还有商品订单表、商品公告表、用户信息表、友情链接表和商品留言表，限于篇幅，这里不再介绍，读者可参见本书附赠光盘中的数据库文件。

26.4　公共文件设计

视频讲解：光盘\TM\Video\第 26 章\公共文件设计.exe

公共模块就是将多个页面都可能使用到的代码写成单独的文件，在使用时只要用 include 或 require 语句将文件包含进来即可。如本系统中的数据库连接、管理和分页类文件，Smarty 模板配置类文件，类的实例化文件，CSS 样式表文件，js 脚本文件等。以前台系统为例，下面给出主要的公共文件，后台的公共文件与前台大同小异。

26.4.1　数据库连接、管理和分页类文件

在数据库连接、管理和分页类文件中，定义 3 个类，分别是 ConnDB 数据库连接类，实现通过 PDO 连接 MySQL 数据库；AdminDB 数据库管理类，使用 PDO 类库中的方法执行对数据库中数据的查询、添加、更新和删除操作；SepPage 分页类，用于对商城中的数据进行分页输出。

（代码位置：光盘\TM\Instance\26\system\system.class.inc.php）

```
<?php
//数据库连接类
class ConnDB{
	var $dbtype;
	var $host;
	var $user;
	var $pwd;
	var $dbname;
	function ConnDB($dbtype,$host,$user,$pwd,$dbname){		//构造方法
		$this->dbtype=$dbtype;
	$this->host=$host;
		$this->user=$user;
```

```
            $this->pwd=$pwd;
            $this->dbname=$dbname;
    }
    function GetConnId(){                                           //实现数据库的连接并返回连接对象
     if($this->dbtype=="mysql" || $this->dbtype=="mssql"){
            $dsn="$this->dbtype:host=$this->host;dbname=$this->dbname";
            }else{
                $dsn="$this->dbtype:dbname=$this->dbname";
            }
            try {
            $conn = new PDO($dsn, $this->user, $this->pwd);   //初始化 PDO 对象，就是创建了数据库连接对象$pdo
                $conn->query("set names utf8");
            return $conn;
            } catch (PDOException $e) {
                die ("Error!: " . $e->getMessage() . "<br/>");
            }
    }
}
//数据库管理类
class AdminDB{
    function ExecSQL($sqlstr,$conn){
            $sqltype=strtolower(substr(trim($sqlstr),0,6));
            $rs=$conn->prepare($sqlstr);                            //准备查询语句
            $rs->execute();                                         //执行查询语句，并返回结果集
            if($sqltype=="select"){
                $array=$rs->fetchAll(PDO::FETCH_ASSOC);             //获取结果集中的所有数据
                if(count($array)==0 || $rs==false)
                    return false;
                else
                    return $array;
            }elseif ($sqltype=="update" || $sqltype=="insert" || $sqltype=="delete"){
                if($rs)
                    return true;
                else
                    return false;
            }
    }
}
//分页类
class SepPage{
    var $rs;
    var $pagesize;
    var $nowpage;
    var $array;
    var $conn;
    var $sqlstr;
    function ShowData($sqlstr,$conn,$pagesize,$nowpage){            //定义方法
            if(!isset($nowpage) || $nowpage=="")                    //判断变量值是否为空
                $this->nowpage=1;                                   //定义每页起始页
            else
                $this->nowpage=$nowpage;
            $this->pagesize=$pagesize;                              //定义每页输出的记录数
            $this->conn=$conn;                                      //连接数据库返回的标识
            $this->sqlstr=$sqlstr;                                  //执行的查询语句
```



```
        $offset=($this->nowpage-1)*$this->pagesize;
        $sql=$this->sqlstr." limit $offset, $this->pagesize";
        $result=$this->conn->prepare($sql);                  //准备查询语句
        $result->execute();                                  //执行查询语句，并返回结果集
        $this->array=$result->fetchAll(PDO::FETCH_ASSOC);    //获取结果集中的所有数据
        if(count($this->array)==0 || $this->array==false)
            return false;
        else
            return $this->array;
    }
    function ShowPage($contentname,$utits,$anothersearchstr,$anothersearchstrs,$class){
        $str="";
        $res=$this->conn->prepare($this->sqlstr);            //准备查询语句
        $res->execute();                                     //执行查询语句，并返回结果集
        $this->array=$res->fetchAll(PDO::FETCH_ASSOC);       //获取结果集中的所有数据
        $record=count($this->array);                         //统计记录总数
        $pagecount=ceil($record/$this->pagesize);            //计算共有几页
        $str.=$contentname." ".$record." ".$utits."  每 页  ".$this->pagesize." 
".$utits." 第 ".$this->nowpage." 页/共 ".$pagecount." 页";
        $str.="    ";
        if($this->nowpage!=1)
            $str.="<a
href=".$_SERVER['PHP_SELF']."?page=1&page_type=".$anothersearchstr."&parameter2=".$anothersearchstr
s." class=".$class.">首页</a>";
        else
            $str.="<font color='#555555'>首页</font>";
        $str.=" ";
        if($this->nowpage!=1)
            $str.="<a
href=".$_SERVER['PHP_SELF']."?page=".($this->nowpage-1)."&page_type=".$anothersearchstr."&parameter2
=".$anothersearchstrs." class=".$class.">上一页</a>";
        else
            $str.="<font color='#555555'>上一页</font>";
        $str.=" ";
        if($this->nowpage!=$pagecount)
            $str.="<a
href=".$_SERVER['PHP_SELF']."?page=".($this->nowpage+1)."&page_type=".$anothersearchstr."&parameter
2=".$anothersearchstrs." class=".$class.">下一页</a>";
        else
            $str.="<font color='#555555'>下一页</font>";
        $str.=" ";
        if($this->nowpage!=$pagecount)
            $str.="<a
href=".$_SERVER['PHP_SELF']."?page=".$pagecount."&page_type=".$anothersearchstr."&parameter2=".$ano
thersearchstrs." class=".$class.">尾页</a>";
        else
            $str.="<font color='#555555'>尾页</font>";
        if(count($this->array)==0 || $this->array==false)
            return "无数据！";
        else
            return $str;
    }
}
?>
```

26.4.2 Smarty 模板配置类文件

在 Smarty 模板配置类文件中配置 Smarty 模板文件、编译文件、配置文件等文件路径。

（代码位置：光盘\TM\Instance\26\system\system.smarty.inc.php）

```
<?php
require("libs/Smarty/Smarty.class.php");                         //包含模板文件
class SmartyProject extends   Smarty{                             //定义类，继承模板类
     function SmartyProject(){                                    //定义方法
          $this->template_dir = "./system/templates/";            //指定模板文件存储位置
          $this->compile_dir = "./system/templates_c/";           //指定编译文件存储位置
          $this->config_dir = "./system/configs/";                //指定配置文件存储位置
          $this->cache_dir = "./system/cache/";                   //指定缓存文件存储位置
     }
}
?>
```

26.4.3 执行类的实例化文件

在 system.inc.php 文件中，通过 require 语句包含 system.smarty.inc.php 和 system.class.inc.php 文件，执行类的实例化操作，并定义返回对象。完成数据库连接类的实例化后，调用其中的 GetConnId()方法连接数据库。

（代码位置：光盘\TM\Instance\26\system\system.inc.php）

```
<?php
require("system.smarty.inc.php");                                        //包含 Smarty 配置类
require("system.class.inc.php");                                         //包含数据库连接和操作类
$connobj=new ConnDB("mysql","localhost","root","111","db_business");    //数据库连接类实例化
$conn=$connobj->GetConnId();                                             //执行连接操作，返回连接标识
$admindb=new AdminDB();                                                  //数据库操作类实例化
$seppage=new SepPage();                                                  //分页类实例化
$usefun=new UseFun();                                                    //使用常用函数类实例化
$smarty=new SmartyProject();                                             //调用 Smarty 模板
function unhtml($params){
   extract($params);
   $text=$content;
   global $usefun;
   return $usefun->UnHtml($text);
}
$smarty->register_function("unhtml","unhtml");                           //注册模板函数
?>
```

26.5 前台首页设计

视频讲解：光盘\TM\Video\第 26 章\前台首页设计.exe

26.5.1 前台首页概述

前台首页一般没有多少实质的技术，主要是加载一些功能模块，如登录模块、导航栏模块、公告栏模



块等，使浏览者能够了解网站内容和特点。首页的重要之处是要合理地对页面进行布局，既要尽可能地将重点模块显示出来，同时又不能因为页面凌乱无序，而让浏览者无所适从、产生反感。本系统前台首页的运行结果如图 26.13 所示。



图 26.13　前台首页运行结果

26.5.2　Smarty 模板页中的框架技术

在前台首页中应用 Switch 语句与 Smarty 模板中的内建函数 include 设计一个框架页面，实现不同功能模块在首页中的展示。

Switch 语句在 PHP 动态文件中使用，根据超链接传递的值，包含不同的功能模块。

Include 标签在 Smarty 模板页中使用，在当前模板页中包含其他模板文件。其语法如下：

```
{include file="file_name " assign=" " var="   "}
```

file 指定包含模板文件的名称；assign 指定一个变量保存包含模板的输出；var 传递给待包含模板的本地参数，只在待包含模板中有效。

26.5.3　前台首页实现过程

（1）创建 index.php 动态页。在 index.php 动态页中，应用 include_once()语句包含相应的文件，应用 switch 语句，以超链接中参数 page 传递的值为条件进行判断，实现在不同页面之间的跳转。index.php 的关键代码如下：

（代码位置：光盘\TM\Instance\26\index.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );          //设置文件编码格式
require("system/system.inc.php");                              //包含配置文件
if(isset($_GET["page"])){
      $page=$_GET["page"];
}else{
      $page="";
```

```
}
include_once("login.php");
include_once("public.php");
include_once("links.php");
switch($page){
        case "hyzx":
                include_once "member.php";
                $smarty->assign('admin_phtml','member.tpl');   //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        case 'allpub':
               include_once 'allpub.php';
                $smarty->assign('admin_phtml','allpub.tpl');      //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        case 'nom':
                include_once 'allnom.php';
                $smarty->assign('admin_phtml','allnom.tpl');      //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        case 'new':
                include_once 'allnew.php';
                $smarty->assign('admin_phtml','allnew.tpl');      //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        case 'hot':
                include_once 'allhot.php';
                $smarty->assign('admin_phtml','allhot.tpl');      //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        case 'shopcar':
                include_once 'myshopcar.php';
                $smarty->assign('admin_phtml','myshopcar.tpl'); //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        case 'settle':
                include_once 'settle.php';
                $smarty->assign('admin_phtml','settle.tpl');      //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        case 'queryform':
                include_once 'queryform.php';
                $smarty->assign('admin_phtml','queryform.tpl');//将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
        default:
                include_once 'newhot.php';
                $smarty->assign('admin_phtml','newhot.tpl');     //将 PHP 脚本文件对应的模板文件名称赋给模板变量
        break;
}
$smarty->display("index.tpl");                                  //指定模板页
?>
```

（2）创建 index.tpl 模板页。在模板文件 index.tpl 中应用 Smarty 的 include 标签调用不同的模板文件，生成静态页面。其关键代码如下：

（代码位置：光盘\TM\Instance\26\system\templates\index.tpl）

```
<table width="850" border="0" cellspacing="0" cellpadding="0">
   <tr>
     <td colspan="2">{include file='top.tpl'}</td>
   </tr>
   <tr>
     <td width="216" align="left" valign="top">
      {include file='login.tpl'}
      {include file='public.tpl'}
```

```
        {include file='links.tpl'}
       </td>
       <td width="634" height="700" align="center" valign="top">
{include file='search.tpl'}
<!--载入模板文件-->{include file=$admin_phtml}</td>
   </tr>
</table>
<table width="850" border="0" cellspacing="0" cellpadding="0">
        <tr>
        <td>{include file='buttom.tpl'}</td>
       </tr>
</table>
```

说明 本系统的功能较多，结构比较复杂，对于初学者来说学起来可能会比较困难。所以，本书将系统中的各个功能模块所涉及的文件（如 PHP、TPL、CSS、JS 等）尽可能都单独实现。读者在学习其中某个模块时，可以将相关的文件统一放到同一个目录下单独测试。

26.6　登录模块设计

视频讲解：光盘\TM\Video\第 26 章\登录模块设计.exe

26.6.1　登录模块概述

用户登录模块是会员功能的窗口。匿名用户虽然也可以访问本网站，但只能进行浏览、查询等简单操作，而会员则可以购买商品，并且能享受超低价格。登录模块包括用户注册、用户登录和找回密码 3 部分，其运行结果如图 26.14 所示。



图 26.14　登录模块运行效果

26.6.2　Ajax 无刷新验证技术

（1）Ajax 技术无刷新验证用户名是否被占用。其关键代码如下：

（代码位置：光盘\TM\Instance\26\js\check.js）

```
/*   form 为传入的表单名称，本段代码为 register 表单   */
function chkname(form){
       /*   如果 name 文本域的信息为空，名为 name1 的 div 标签显示如下信息   */
       if(form.name.value==""){
              name1.innerHMRL="<font color=#FF0000>请输入用户名！</font>";
       }else{
              /*   否则获取文本域的值   */
              var user = form.name.value;
              /*   生成 url 链接，将 user 的值传到 chkname.php 页进行判断   */
              var url = "chkname.php?user="+user;
              /*   使用 xmlhttprequest 技术运行页面   */
              xmlhttp.open("GET",url,true);
              xmlhttp.onreadystatechange = function(){
              if(xmlhttp.readyState == 4){
                            /*   根据不同的返回值，在 div 标签中输出不同信息   */
```

```
                        var msg = xmlhttp.responseText;
                        if(msg == '3'){
                            name1.innerHMRL="<font color=#FF0000>用户名被占用！ </font>";
                            return false;
                        }else if(msg == '2'){
                            name1.innerHMRL="<font color=green>恭喜您，可以注册!</font>";
                        /*  如果用户名正确，则将隐藏域的值改为 yes */
                            form.c_name.value = "yes";
                        }else{
                            name1.innerHMRL="<font color=green>未知错误</font>";
                        }
                    }
                }
                xmlhttp.send(null);
        }
}
```

在该函数中调用 chkname.php 页，该页在会员登录时也会被调用，所以这里分两种情况，有密码和无密码。无密码为注册验证，当没有返回结果时，说明该用户名可用；而有密码为登录验证，和无密码相反，只有查询记录存在时，才允许登录，并将用户名和用户 ID 存储到 session 中。chkname.php 页面的代码如下：

（代码位置：光盘\TM\Instance\26\chkname.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );              //设置文件编码格式
require("system/system.inc.php");                                 //包含配置文件
$reback = '0';
$sql = "select * from tb_user where name='".$_GET['user']."'";
if(isset($_GET['password'])){
    $sql .= " and password = '".md5($_GET['password'])."'";
}
$rst = $admindb->ExecSQL($sql,$conn);
if($rst){
    /*  登录  */
    if($rst[0]['isfreeze'] != 0){
        $reback = '3';
    }else{
        $_SESSION['member'] = $rst[0]['name'];
        $_SESSION['id'] = $rst[0]['id'];
        $reback = '2';
    }
}else{
    $reback = '1';
}
echo $reback;
?>
```

（2）GD2 函数库生成验证码。其关键代码如下：

（代码位置：光盘\TM\Instance\26\yzm.php）

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );              //设置文件编码格式
srand((double)microtime()*1000000);                               //生成随机数
$im=imagecreate(60,30);                                           //创建画布
$black=imagecolorallocate($im,0,0,0);                             //定义背景
$white=imagecolorallocate($im,255,255,255);                       //定义背景
$gray=imagecolorallocate($im,200,200,200);                        //定义背景
imagefill($im,0,0,$gray);                                         //填充颜色
```

```
for($i=0;$i<4;$i++){                                        //定义 4 位随机数
 $str=mt_rand(3,20);                                        //定义随机字符所在位置的 Y 坐标
 $size=mt_rand(5,8);                                        //定义随机字符的字体
 $authnum=substr($_GET['num'],$i,1);                        //获取超链接中传递的验证码
 imagestring($im,$size,(2+$i*15),$str,$authnum,imagecolorallocate($im,rand(0,130),rand(0,130),rand(0,130)));
}                                                           //水平输出字符串
for($i=0;$i<200;$i++){                                      //执行 for 循环，为验证码添加模糊背景
  $randcolor=imagecolorallocate($im,rand(0,255),rand(0,255),rand(0,255));    //创建背景
  imagesetpixel($im,rand()%70,rand()%30,$randcolor);                         //绘制单一元素
}
imagepng($im);                                                               //生成 PNG 图像
imagedestroy($im);                                                           //销毁图像
?>
```

26.6.3　用户注册

用户注册页面的主要功能是新用户注册。如果信息输入完整而且符合要求，则系统会将该用户信息保存到数据库中，否则显示错误原因，以便用户改正。用户注册页面的运行结果如图 26.15 所示。

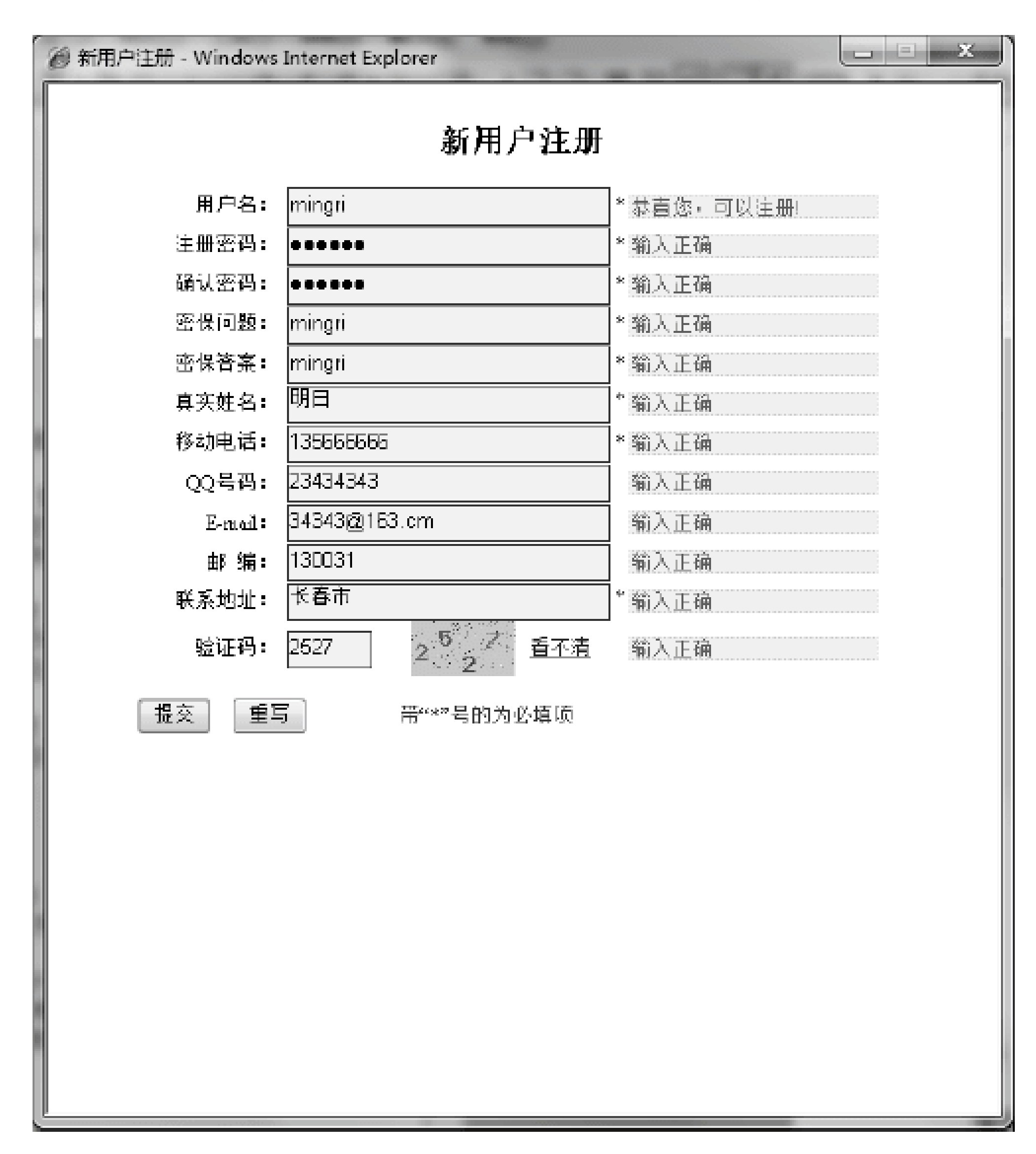


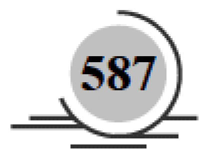
图 26.15　注册模块页面

（1）创建 register.tpl 模板文件，编写用户注册页面。其中包含两个 JS 脚本文件 createxmlhttp.js 和 check .js。其中，createxmlhttp.js 是 Ajax 的实例化文件，而 check.js 对用户注册信息进行验证，并且返回验证结果。

（2）创建 register.php 动态 PHP 文件，加载模板。register.php 文件的代码如下：

（代码位置：光盘\TM\Instance\26\register.php）

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );        //设置文件编码格式
require("system/system.inc.php");                           //包含配置文件
$smarty->assign('title','新用户注册');
$smarty->display('register.tpl');
?>
```



（3）创建 reg_chk.php 文件，获取表单中提交的数据，将数据存储到指定的数据表中。reg_chk.php 文件的代码如下：

（代码位置：光盘\TM\Instance\26\register.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );                    //设置文件编码格式
require("system/system.inc.php");                                       //包含配置文件
	$name = $_POST['name'];
	$password = md5($_POST['pwd1']);
	$question = $_POST['question'];
	$answer = $_POST['answer'];
	$realname = $_POST['realname'];
	$card = $_POST['card'];
	$tel = $_POST['tel'];
	$phone = $_POST['phone'];
	$Email = $_POST['email'];
	$QQ = $_POST['qq'];
	$code = $_POST['code'];
	$address = $_POST['address'];
	$addtime = date("Y-m-d H:i:s");
	$sql="insertinto
tb_user(name,password,question,answer,realname,card,tel,phone,Email,QQ,code,address,addtime,isfreeze,sh
opping,consume)" ;
	$sql .= " values ('$name', '$password', '$question', '$answer', '$realname', '$card', '$tel', '$phone', '$Email',
'$QQ', '$code', '$address','$addtime','0','','00.00')";
	$rst= $admindb->ExecSQL($sql,$conn);
	if($rst){
		$_SESSION['member'] = $name;
		echo "<script>top.opener.location.reload();alert('注册成功');window.close();</script>";
	}else{
		echo '<script>alert(\'添加失败\');history.back;</script>';
	}
?>
```

（4）创建“用户注册”超链接。当用户单击前台的 注册 时，系统会调用 JS 文件的 onclick 事件，弹出注册窗口。其代码如下：

（代码位置：光盘\TM\Instance\26\system\templates\login.tpl）

```
<a href="#" id="login" onclick="reg()"><img src="images/check.JPG" width="59" height="23" border="0" /></a>
```

这里使用到的 JS 文件为 js/login.js，调用的函数为 reg()。该函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\login.js）

```
function reg(){
window.open("register.php", "_blank", "width=600,height=650",false);      //弹出窗口
}
```

26.6.4　用户登录

用户登录模块的运行结果如图 26.16 所示，需要输入用户名、密码和验证码。



图 26.16　用户登录页面

（1）创建模板文件 login.tpl，完成用户登录表单的设计。在该页面中当单击 Submit 按钮时，系统将调用 lg()函数对用户登录提交信息进行验证。lg()函数包含在 js/login.js 脚本文件内。其代码如下：

（代码位置：光盘\TM\Instance\26\js\login.js）



```
//JavaScript Document
function lg(form){
	if(form.name.value==""){
		alert('请输入用户名');
		form.name.focus();
		return false;
	}
	if(form.password.value == "" || form.password.value.length < 6){
		alert('请输入正确密码');
		form.password.focus();
		return false;
	}
	if(form.check.value == ""){
		alert('请输入验证码');
		form.check.focus();
		return false;
	}
	if(form.check.value != form.check2.value){
		form.check.select();
		code(form);
		return false;
	}
	var user = form.name.value;
	var password = form.password.value;
	var url = "chkname.php?user="+user+"&password="+password;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = function(){
	if(xmlhttp.readyState == 4){
			var msg = xmlhttp.responseText;
			if(msg == '1'){
				alert('用户名或密码错误!!');
				form.password.select();
				form.check.value = '';
				code(form);
				return false;
			}if(msg == "3"){
				alert("该用户被冻结，请联系管理员");
				return false;
			}else{
				alert('欢迎光临');
				location.reload();
			}
		}
	}
	xmlhttp.send(null);
	return false;
}
//显示验证码
function yzm(form){
	var num1=Math.round(Math.random()*10000000);
	var num=num1.toString().substr(0,4);
	document.write("<img name=codeimg width=65 heigh=35 src='yzm.php?num="+num+"'>");
	form.check2.value=num;
}
//刷新验证码
function code(form){
	var num1=Math.round(Math.random()*10000000);
	var num=num1.toString().substr(0,4);
```

```
        document.codeimg.src="yzm.php?num="+num;
        form.check2.value=num;
}
//注册
function reg(){
window.open("register.php", "_blank", "width=600,height=650",false);
}
//找回密码
function found() {
window.open("found.php","_blank","width=350 height=240",false);
}
```

用户名和密码是在 chkname.php 页面中被验证的。chkname.php 在 26.6.2 节中已经介绍过，这里不再重复了。

（2）创建用户信息模板文件 info.tpl。用户登录成功后，在原登录框位置将显示用户信息，用户可以通过“会员中心”对自己的信息做修改，也可以单击“查看购物车”超链接查看购物车商品；当用户离开时可以单击“安全离开”超链接。用户信息模块的主要代码如下：

（代码位置：光盘\TM\Instance\26\system\templates\info.tpl）

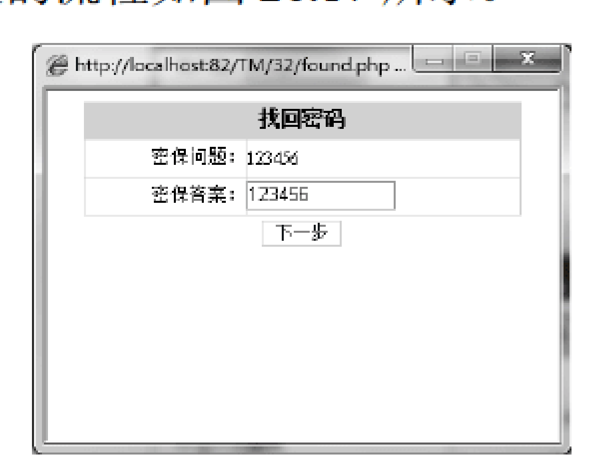
```
<!--显示当前登录用户名-->
欢迎您：{$member}
<!--会员中心超链接-->
<a href="?page=hyzx" id="info" class="lk">会员中心</a>
<!--查看购物车-->
<a href="?page=shopcar" class="lk">查看购物车</a>
<!--安全离开-->
<a onclick="javascript:logout()" style="cursor:hand" id="info">安全离开</a>
```

26.6.5　找回密码

登录模块的最后一个部分就是找回密码。找回密码是根据用户在填写资料时所填写的密保问题和密保答案来实现的。当用户单击“找回密码”超链接时，首先提示用户输入要找回密码的会员名称，然后根据密保问题填写密保答案，最后重新输入密码。找回密码模块的流程如图 26.17 所示。



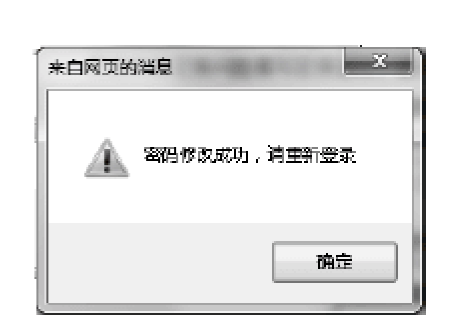
第一步：输入会员名称



第二步：输入密保答案



第三步：输入新密码



第四步：密码修改成功

图 26.17　找回密码流程图



1．创建模板文件

虽然找回密码需要 4 个步骤，但实际上每个步骤使用的都是相同的模板文件和 JS 文件，只是被调用的表单和 js 函数略有差别。这里根据不同的文件来分别进行介绍。

该模板文件一共包含了 3 个表单，分别代表了 3 个步骤。其核心代码如下：

（代码位置：光盘\TM\Instance\26\system\templates\found.tpl）

```
<!--  载入两个 JS 脚本文件  -->
<script language="javascript" src="js/createxmlhttp.js"></script>
<script language="javascript" src="js/found.js"></script>
<!--  第 1 个 div 标签  -->
<div id="first">
<table width="200" border="0" cellspacing="0" cellpadding="0">
<form id="foundname" name="found" method="post" action="#">
  <tr><td> 找回密码</td></tr>
  <tr><td>会员名称：</td>
     <!--  text 文本域，用于输入要找回密码的会员名称  -->
<td><input id="user" name="user" type="text" class="txt"></td>
</tr>
  <tr><td>
<!--  单击“下一步”按钮，能触发 onclick 事件来调用 chkname 函数  -->
<input id = " next1 " name = " next1 " type = " button " class = " btn " value = " 下一步 " onClick = " return chkname ( foundname ) "/></td></tr>
</form>
</table>
</div>
<!--  第 2 个 div 标签，样式为隐藏  -->
<div id="second" style="display:none;">
<table>
<form id="foundanswer" name="found" method="post" action="#">
  <tr><td > 找回密码</td></tr>
  <tr><td>密保问题：</td>
<!--  用于显示密保问题的 div 标签  -->
    <td <div id="question"></div></td></tr>
  <tr><td>密保答案：</td>
<!--  文本域，用于填写密保答案  -->
    <td ><input id="answer" name="answer" type="text" class="txt" /></td></tr>
  <tr>
<!--  单击“下一步”按钮，用来触发 onclick 事件，并调用 chkanswer()函数  -->
    <td><input id = " next2 " name = " next2 " type=" button " class=" btn " value ="  下一步  " onClick = " return chkanswer ( foundanswer ) "></td>
    </tr>
</form>
</table>
</div>
<!--  第 3 个 div 标签，样式也为隐藏，作用是修改密码  -->
<div id='third' style="display:none;">
<table>
<form id="modifypwd" name="found" method="post" action="#">
  <tr><td> 输入密码</td></tr>
  <tr><td>输入密码：</td>
    <td><input id="pwd1" name="pwd1" type="password" class="txt"></td></tr>
  <tr><td>确认密码：</td>
    <td><input id="pwd2" name="pwd2" type="password" class="txt" /></td>
  </tr>
```

```
    <tr>
<!--  单击“完成”按钮，调用 ckpwd()函数  -->
      <td><input id = " mod " name = " mod " type = " button " class = " btn " value = " 完成 " onClick = " return chkpwd (modifypwd) "></td>
      </tr>
</form>
</table>
</div>
```

可以看出，在上述 3 个表单中，只有一个表单默认情况下是显示的，其他则为隐藏。只有通过调用不同的 js 函数，才可以对其他表单进行操作。

2. 创建 JS 脚本文件

found.js 脚本文件包含 3 个函数：chkname()、chkanswer()和 chkpwd()。其中，chkname()函数的作用是检查用户输入的会员名称，如果存在，则使用 xmlhttp 对象去调用生成的 url 进行处理判断。如果该用户存在，则隐藏当前表单，并显示下一个表单，最后输出密保问题。chkname()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\found.js）

```
function chkname(form){
	var user = form.user.value;
	if(user == ''){
		alert('请输入用户名');
		form.user.focus();
		return false;
	}else{
		var url = "foundpwd.php?user="+user;
		xmlhttp.open("GET",url,true);
		xmlhttp.onreadystatechange = function(){
		if(xmlhttp.readyState == 4){
				var msg = xmlhttp.responseText;
				if(msg == '0'){
					alert('没有该用户，请重新查找!');
					form.user.select();
					return false;
				}else{
					document.getElementById('first').style.display = 'none';
					document.getElementById('second').style.display = '';
					document.getElementById('question').innerHTML = msg;
				}
			}
		}
		xmlhttp.send(null);
	}
}
```

其他两个函数也使用 xmlhttprequest 对象，实现方法相差无几，不同之处就是对返回值的处理。chkanswer()函数隐藏当前表单，显示下一个表单。chkanswer()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\found.js）

```
function chkanswer(form) {
	var user = document.getElementById('user').value;
	var answer = form.answer.value;
	if(answer == ''){
		alert('请输入提示问题');
		form.answer.focus();
		return false;
	}else{
```



```
                var url = "foundpwd.php?user="+user+"&answer="+answer;
                xmlhttp.open("GET",url,true);
                xmlhttp.onreadystatechange = function(){
                        if(xmlhttp.readyState == 4){
                                var msg = xmlhttp.responseText;
                                if(msg == '0'){
                                        alert('问题回答错误');
                                        form.answer.select();
                                        return false;
                                }else{
                                        document.getElementById('second').style.display = 'none';
                                        document.getElementById('third').style.display = '';
                                }
                        }
                }
                xmlhttp.send(null);
        }
}
```

而 chkpwd()函数则提示用户操作状态，如果成功，则关闭当前页。ckpwd()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\found.js）

```
function chkpwd(form){
        var user = document.getElementById('user').value;
        var pwd1 = form.pwd1.value;
        var pwd2 = form.pwd2.value;
        if(pwd1 == ''){
                alert('请输入密码');
                form.pwd1.focus();
                return false;
        }
        if(pwd1.length < 6){
                alert('密码输入错误');
                form.pwd1.focus();
                return false;
        }
        if(pwd1 != pwd2){
                alert('两次密码不相等');
                form.pwd2.select();
                return false;
        }
        var url = "foundpwd.php?user="+user+"&password="+pwd1;
        xmlhttp.open("GET",url,true);
        xmlhttp.onreadystatechange = function(){
                if(xmlhttp.readyState == 4){
                        var msg = xmlhttp.responseText;
                        if(msg == '1'){
                                alert('密码修改成功，请重新登录');
                                window.close();
                        }else{
                                alert(msg);
                        }
                }
        }
        xmlhttp.send(null);
}
```

3．创建数据处理文件

foundpwd.php 文件的功能是根据用户输入信息来检测数据表中的数据，并根据不同的输入信息返回不同的结果。该文件的代码如下：

（代码位置：光盘\TM\Instance\26\foundpwd.php）

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );        //设置文件编码格式
require("system/system.inc.php");                           //包含配置文件 $smarty->assign('title','找回密码');
$reback = '0';                                                  //设置变量初始值
if(!isset($_GET['answer']) && !isset($_GET['password'])){      //判断变量是否存在
	$namesql = "select * from tb_user where name = '".$_GET['user']."'";
	$namerst = $admindb->ExecSQL($namesql,$conn);           //查询用户名是否存在
	if($namerst){
		$question = $namerst[0]['question'];
		$reback = $question;
	}
}else if(isset($_GET['answer'])){
	$answersql = "select * from tb_user where name = '".$_GET['user']."' and answer = '".$_GET['answer']."'";
	$answerrst = $admindb->ExecSQL($answersql,$conn);
	if($answerrst){
		$reback = '1';
	}
}else if(isset($_GET['password'])){
	$sql="update tb_user set password='".md5($_GET['password'])."' where name='".$_GET['user']."'";
	$rst = $admindb->ExecSQL($sql,$conn);
	if($rst){
		$reback = '1';                                      //为模板变量赋值
	}
}
echo $reback;                                               //输出返回结果
?>
```

4．加载模板页

因为所有登录模块的模板都不需要或者只需要传递一两个变量，所以 PHP 加载页的内容比较简单。找回密码页面的代码如下：

（代码位置：光盘\TM\Instance\26\found.php）

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );        //设置文件编码格式
require("system/system.inc.php");                           //包含配置文件 $smarty->assign('title','找回密码');
$smarty->display('found.tpl');
?>
```

26.7 会员信息模块设计

视频讲解：光盘\TM\Video\第 26 章\会员信息模块设计.exe

26.7.1 会员信息模块概述

用户登录后，即可看到会员信息模块。在这里，可以进行查看或修改个人信息及密码、查看购物车和



安全退出等操作。本节只对会员信息模块中的“会员中心”和“安全退出”进行讲解，关于“查看购物车”将在商品模块中进行介绍。会员信息模块的运行效果如图 26.18 所示。

图 26.18　会员中心

26.7.2　会员信息查询技术

在会员信息模块中，以 SESSION 变量中存储的用户名称为条件，从会员信息表中查询出会员信息，并且将会员信息存储到模板变量中，最后在模板页中输出会员信息。member.php 的代码如下：

（代码位置：光盘\TM\Instance\26\member.php）

```
<?php
/*	查找用户资料	*/
if(isset($_SESSION['member'])){
	$sql = "select * from tb_user where name = '".$_SESSION['member']."'";
	$arr = $admindb->ExecSQL($sql,$conn);
	if(isset($_GET['action']) && $_GET['action'] == 'modify'){
		$smarty->assign('check',"find");
		$smarty->assign('pwdarr',$arr);
	}else{
		$smarty->assign('check',"notfind");
		$smarty->assign('pwdarr',$arr);
	}
}
?>
```

member.php 文件中查询出的数据是会员信息模板功能实现的根本。

26.7.3　会员中心

当单击“会员中心”超链接时，会回传给当前页一个 page 值，当前页根据这个 page 值来载入 member.php 文件。

1．创建 PHP 页面

与登录模块设计不同，本节首先来创建 PHP 页面。因为该模块中的模板需要使用数据库中的数据及一些动态信息，这些都需要在 PHP 页中先行获取及处理，然后再传给模板页。会员中心页面的代码请参考技术分析中的内容。

2．创建模板页

该模块包括查看信息模板及修改密码模板，都存储于 member.tpl 模板文件中。

（代码位置：光盘\TM\Instance\26\system\templates\member.tpl）

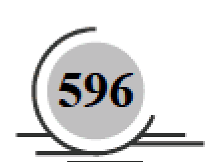
```
<link rel="stylesheet" href="css/member.css" />
<script language="javascript" src="js/member.js"></script>
{if $check=="find" }
<p align="left">{$smarty.session.member}&gt;&gt;&gt;<a href='?page=hyzx' id="mem">查看信息</a>&gt;&gt;
&gt;<a href='?page=hyzx&action=modify' id="mem">修改密码</a></p>
<table id="member" width="300" border="0" cellpadding="0" cellspacing="0">
<form id="member" name="member" method="post" action="modify_pwd_chk.php" onSubmit="return
pwd(member)">
  <tr>
    <td height="25" colspan="2" align="center" valign="middle" id="first"><font color="#f0f0f0">修改密码
</font></td>
  </tr>
……//省略了部分代码
  <tr>
    <td height="30" colspan="2" align="center" valign="middle"><input id="enter" name="enter" type="submit"
value="修改" /></td>
  </tr>
</form>
</table>
{else}
<p align="left">{$smarty.session.member}&gt;&gt;&gt;<a href='?page=hyzx' id="mem">查看信息</a>&gt;&gt;
&gt;<a href='?page=hyzx&action=modify' id="mem">修改密码</a></p>
{section name=pwd_id loop=$pwdarr}
<table id='member' width="500" border="0" cellpadding="0" cellspacing="0">
<form id="member" name="member" method="post" action="modify_info_chk.php" onSubmit="return
mem(member)" >
  <tr>
    <td height="25" colspan="2" align="center" valign="middle" id="first"><font
color="#f0f0f0">{$pwdarr[pwd_id].name}信息（不可更改信息）</font></td>
  </tr>
  <tr>
    <td width="25%" height="25" align="right" valign="middle" id="left"> 会员编号：</td>
    <td height="25" align="left" valign="middle" id="right"> {$pwdarr[pwd_id].id}</td>
  </tr>
……//省略了部分代码
  <tr>
    <td height="30" colspan="2" align="center" valign="middle"><input name="enter" type="submit" id="enter"
value="修改" />    <input name="reset" type="reset" id="reset" value="重置" /></td>
  </tr>
</form>
</table>
{/section}
{/if}
```

3．创建脚本文件

该模块的脚本文件和用户注册模块类似，都是对信息的合法性进行验证，如信息是否为空、是否符合规范等，这里就不再赘述了。

4．创建处理页

当信息验证通过后，系统将跳转到处理页进行信息处理。本模块处理页分信息修改和密码修改两个页



面。首先介绍信息修改页，其代码如下：

（代码位置：光盘\TM\Instance\26\modify_info_chk.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
require("system/system.inc.php");                                   //包含配置文件
$sql = "update tb_user set realname='".$_POST['realname']."',card='".$_POST['card']."',tel='".$_POST['tel']."',
phone= '".$_POST['phone']."',Email='".$_POST['email']."',QQ='".$_POST['qq']."',code='".$_POST['code']."',
address='".$_POST['address']."' where id = '".$_POST['userid']."'";
$arr = $admindb->ExecSQL($sql,$conn);
if($arr)
    echo "<script>alert('修改成功');location=('index.php');</script>";
else
    echo "<script>alert('修改失败');history.go(-1);</script>";
?>
```

与信息修改页的操作流程十分类似，只是更新的数组要小得多，只有一个字段。修改密码页的代码如下：

（代码位置：光盘\TM\Instance\26\modify_pwd_chk.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
require("system/system.inc.php");                                   //包含配置文件
$sql="select * from tb_user where name = '".$_SESSION['member']."' and password='".md5($_POST['old'])."' ";
$arr = $admindb->ExecSQL($sql,$conn);                               //判断用户名和密码是否正确
if($arr){
    $sql = "update tb_user set password='".md5($_POST['new1'])."' where name = '".$_SESSION['member']."'
and password='".md5($_POST['old'])."' ";                            //更新密码
    $arr = $admindb->ExecSQL($sql,$conn);
    echo "<script>alert('密码修改成功！'); window.location.href='index.php';</script>";
}else{
    echo "<script>alert('密码修改失败！'); window.location.href='index.php';</script>";
}
?>
```

26.7.4　安全退出

当用户需要离开网站时，可以单击“安全退出”超链接来调用 logout()函数，当用户确认退出后，则跳转到 logout 页面，销毁 session 并回到首页。安全退出所涉及的页面及代码如下：

（代码位置：光盘\TM\Instance\26\js\info.js）

```
function logout(){
    if(confirm("确定要退出登录吗？ ")){                  //输出选择框，用户可以单击“确认”或“取消”按钮
        window.open('logout.php','_parent','',false);    //如果用户确认退出，则打开 logout.php 页
    }else
     return false;
}
```

（代码位置：光盘\TM\Instance\26\logout.php）

```
<?php
    session_start();
    header ( "Content-type: text/html; charset=UTF-8" );        //设置文件编码格式
    session_destroy();
    echo '<script>alert(\'用户已安全退出!\');location=(\'index.php\');</script>';
?>
```

26.8 商品展示模块设计

视频讲解：光盘\TM\Video\第 26 章\商品展示模块设计.exe

26.8.1 商品展示模块概述

本系统为用户提供了不同的商品展示方式，包括推荐商品、最新商品、热门商品等，能够使消费者有目的地选购商品。每个展示方式中都有商品详细信息的显示，为用户购买商品提供可靠的依据。本系统商品显示模块的运行结果如图 26.19 所示。

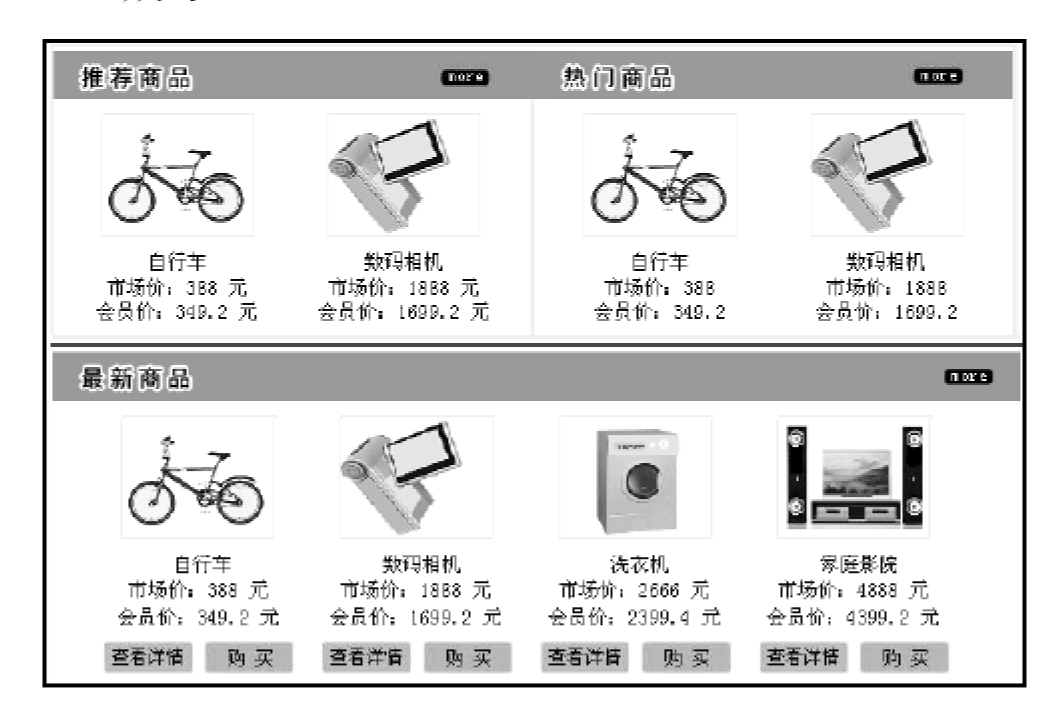


图 26.19　商品展示模块页面

因为推荐商品、最新商品和热门商品的实现方法和过程基本相同，所以本节只讲解推荐商品模块。其他功能的相关代码可参见光盘中的源程序。

26.8.2 分页技术

商品显示功能实现的关键就是如何从数据库中读取商品信息，如何完成数据的分页显示。首先，包含类的实例化文件。然后，判断分页变量 page 的值是否存在。接着，定义 SQL 语句，对查询结果进行降幂排列，并且设置每页显示 3 条记录。再调用分页类中的方法完成数据的分页读取和输出。其代码如下：

（代码位置：光盘\TM\Instance\26\allnom.php）

```
<?php
include_once("system/system.inc.php");                    //包含类的实例化文件
if(isset($_GET['page'])){                                  //判断当前页变量的值
      $page=$_GET['page'];
}else{
      $page=1;
}
$rst1 = $seppage->ShowData("select * from tb_commo where isnom = 1 order by isnom,id desc",$conn,3,$page);
                                                           //调用分页类方法
$smarty->assign("nomarr",$rst1);
$smarty->assign('rst1_page',$seppage->ShowPage("产品","个",$_GET['page_type'],"","a"));
                                                           //输出分页超链接
$smarty->assign('title','推荐商品');
?>
```

最后，定义模板文件，通过 section 语句循环输出存储在模板变量中的数据，并且输出分页超链接。

26.8.3　商品展示模块的实现过程

在技术分析中已经对商品显示所使用的技术、方法进行概述，下面就介绍一下它具体的过程。

（1）创建 allnom.php 文件，从数据库中读取出推荐商品的数据，并将数据存储到模板变量中，其代码可以参考技术分析。

（2）创建 allnom.tpl 模板页，应用 section 语句输出商品信息，并添加相应的操作按钮或链接。模板页中一共有两个事件：查看商品详细信息和放入购物车。

☑ 当单击“查看详情”按钮时，将触发 onclick 事件，并将调用 openshowcommo()函数，同时，商品 id 会作为函数的唯一参数被传递进去。

☑ 当单击“购买”按钮时，同样会触发 onclick 事件，并调用 buycommo()函数，唯一的参数也是商品的 id。

商品模板页面的代码如下：

（代码位置：光盘\TM\Instance\26\system\templates\allnom.tpl）

```
<table width="636" border="0" align="center" cellspacing="0" cellpadding="0">
  <tr>
    <td height="33" align="left" valign="middle" background="images/shop_07.gif"> </td>
  </tr>
  <tr>
    <td height="132" align="left" valign="middle">
   {section name=nom_id loop=$nomarr}
    <table width="636" height="134" border="0" cellspacing="0" cellpadding="0">
     <tr>
          <td width="145" rowspan="5" align="center" valign="middle"><img src="{$nomarr[nom_id].pics}" width="90" height="100" alt="{$nomarr[nom_id].name}" style="border: 1px solid #f0f0f0;" /></td>
              <td height="35">商品名称：{$nomarr[nom_id].name}</td>
              <td width="156" height="35">商品类别：{$nomarr[nom_id].class}</td>
          <td width="157">商品型号：{$nomarr[nom_id].model}</td>
     </tr>
     <tr>
       <td height="23">商品品牌：{$nomarr[nom_id].brand}</td>
           <td height="23" colspan="2">商品产地：{$nomarr[nom_id].area}</td>
     </tr>
     <tr>
       <td width="178" height="23">剩余数量：{$nomarr[nom_id].stocks}</td>
           <td colspan="2">销售数量：{$nomarr[nom_id].sell}</td>
     </tr>
     <tr>
       <td height="23">市场价：<font color="red">{$nomarr[nom_id].m_price} 元</font></td>
           <td height="23" colspan="2">上市日期：{$nomarr[nom_id].addtime}</td>
     </tr>
     <tr>
       <td height="30">会员价格：<font color="#FF0000">{$nomarr[nom_id].v_price} 元</font></td>
           <td height="30" colspan="2" align="center" valign="middle"><input id="allshow" name="allshow" type="button" value="" class="showinfo" onclick="openshowcommo({$nomarr[nom_id].id})" />  <input id="buy" name="buy" type="button" value="" class="buy" onclick="return buycommo({$nomarr[nom_id].id})" /></td>
     </tr>
    </table>
    <hr style="border: 1px solid #f0f0f0;" />
    {/section}
```

```
        </td>
    </tr>
    <tr>
        <td height="25">{$rst1_page}</td>
    </tr>
</table>
```

（3）创建 showcommo.js 脚本文件。当单击“查看商品”按钮时，系统会弹出一个新的页面，并显示商品的详细信息；当单击“购买”按钮时，该商品将会被放到当前用户的购物车中，如果没有登录用户或商品已添加，则会提示错误信息。JS 脚本文件的代码如下：

（代码位置：光盘\TM\Instance\26\js\showcommo.js）

```
/*    查看商品信息函数，将打开一个新页面    */
function openshowcommo(key){
    open('showcommo.php?id='+key,'_blank','width=560 height=300',false);
}
/*    将购买商品添加到购物车中，将在下节中讲解    */
function buycommo(key){
    …
}
```

26.9 购物车模块设计

视频讲解：光盘\TM\Video\第 26 章\购物车模块设计.exe

26.9.1 购物车模块概述

购物车是电子商务平台中非常关键的一个功能模块。购物车的主要功能是保留用户选择的商品信息，用户可以在购物车内设置选购商品的数量、显示选购商品的总金额，还可以清除选择的全部商品信息，重新选择商品信息。购物车页面的运行结果如图 26.20 所示。

我的购物车						
	商品名称	购买数量	市场价格	会员价格	折扣率	合计
□	自行车	3	388	349.2	9	1047.6
□	数码相机	5	1888	1699.2	9	8496
□	洗衣机	1	2666	2399.4	9	2399.4
□	家庭影院	1	4888	4399.2	9	4399.2

全选 反选 删除选择 继续购物 去收银台 共计：16342.2 元

图 26.20 购物车页面

购物车模块主要实现添加商品、删除商品和更改数量等操作。

26.9.2 购物车中商品添加技术

购物车功能的实现最关键的部分就是如何将商品添加到购物车，如果不能完成商品的添加，那么购物车中的其他操作都没有任何意义。

（1）在商品显示模块中，单击商品中的“购买”按钮，将商品放到购物车中，并进入到购物车页面。单击“购买”按钮调用 buycommo()函数，购买商品的 id 是该函数的唯一参数，在 buycommo()函数中通过 xmlhttp 对象调用 chklogin.php 文件，并根据回传值做出相应处理。buycommo()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\showcommo.js）

```
/*
*添加商品，同时检查用户是否登录、商品是否重复等
*/
function buycommo(key){
```



```
	/*	根据商品 ID，生成 url	*/
	var url = "chklogin.php?key="+key;
	/*	使用 xmlhttp 对象调用 chklogin.php 页	*/
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = function(){
		if(xmlhttp.readyState == 4){
				var msg = xmlhttp.responseText;
				/*	用户没有登录	*/
				if(msg == '2'){
					alert('请您先登录');
					return false;
				}else if(msg == '3'){
				/*	商品已添加	*/
					alert('该商品已添加');
					return false;
				}else{
				/*	显示购物车	*/
					location='index.php?page=shopcar';
				}
			}
	}
	xmlhttp.send(null);
```

（2）在 chklogin.php 文件中将商品添加到购物车中。chklogin.php 页的代码如下：

（代码位置：光盘\TM\Instance\26\chklogin.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );			//设置文件编码格式
require("system/system.inc.php");					//包含配置文件
/**
	*	1 表示添加成功
	*	2 表示用户没有登录
	*	3 表示商品已添加过
	*	4 表示添加时出现错误
	*	5 表示没有商品添加
*/
$reback = '0';
if(empty($_SESSION['member'])){
	$reback = '2';
}else{
	$key = $_GET['key'];
	if($key == ''){
		$reback = '5';
	}else{
		$boo = false;
		$sqls = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
		$shopcont = $admindb->ExecSQL($sqls,$conn);
		if(!empty($shopcont[0]['shopping'])){
			$arr = explode('@',$shopcont[0]['shopping']);
			foreach($arr as $value){
				$arrtmp = explode(',',$value);
				if($key == $arrtmp[0]){
					$reback = '3';
					$boo = true;
					break;
```

```
                }
            }
            if($boo == false){
                $shopcont[0]['shopping'] .= '@'.$key.',1';
                $update = "update tb_user set shopping='".$shopcont[0]['shopping']."' where name = '".$_SESSION['member']."'";
                $shop = $admindb->ExecSQL($update,$conn);
                if($shop){
                    $reback = 1;
                }else{
                    $reback = '4';
                }
            }
        }else{
            $tmparr = $key.",1";
            $updates = "update tb_user set shopping='".$tmparr."' where name = '".$_SESSION['member']."'";
            $result = $admindb->ExecSQL($updates,$conn);
            if($result){
                $reback = 1;
            }else{
                $reback = '4';
            }
        }
    }
}
echo $reback;
?>
```

通过分析上述代码可知，shopping 字段保存的是购物车中的商品信息，一条商品信息包括两部分，即商品 id 和商品数量，其中商品数量默认为 1。两部分之间使用逗号（,）分隔，如果添加多个商品，则每个商品之间使用“@”分隔。

成功完成商品的添加操作后，即可进入到购物车页面，执行其他操作。

26.9.3 购物车展示

购物车页面分 PHP 代码页和 Smarty 模板页。在 PHP 代码页中，首先读取 tb_user 数据表中 shopping 字段的内容，如果字段为空，则输出“暂无商品”；如果数据库中有数据，则循环输出数据，并将商品信息保存到数组中，再传给模板页。购物车页面的代码如下：

（代码位置：光盘\TM\Instance\26\myshopcar.php）

```
<?php
$select = "select id,shopping from tb_user where name ='".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($select,$conn);
if($rst[0]['shopping']==""){
    echo "<script>alert('购物车中暂时没有商品!');window.location.href='index.php';</script>";
}
$commarr = array();
foreach($rst[0] as  $value){
    $tmpnum = explode('@',$value);
    $shopnum = count($tmpnum);                              //商品类数
    $sum = 0;
    foreach($tmpnum as $key => $vl){
        $s_commo = explode(',',$vl);
        $sql2 = "select id,name,m_price,fold,v_price from tb_commo";
```

```
            $commsql = $sql2." where id = ".$s_commo[0];
            $arr = $admindb->ExecSQL($commsql,$conn);
            @$arr[0]['num'] = $s_commo[1];
            @$arr[0]['total'] = $s_commo[1]*$arr[0]['v_price'];
            $sum += $arr[0]['total'];
            $commarr[$key] = $arr[0];
      }
}
$smarty->assign('shoparr',$shopnum);
$smarty->assign('commarr',$commarr);
$smarty->assign('sum',$sum);
?>
```

商品的模板页不仅要负责用户购买商品信息的输出，而且还要提供可以对商品进行修改、删除等操作的事件接口。模板页的代码如下：

（代码位置：光盘\TM\Instance\26\myshopcar.php）

```
<table border="0" cellspacing="0" cellpadding="0" align="center">
<form id="myshopcar" name="myshopcar" method="post" action="#">
  <tr>
    <td height="30" colspan="7" align="center" valign="middle" class="first">我的购物车</td>
  </tr>
  <tr>
    <td width="35" height="25" align="center" valign="middle" class="left"> </td>
    <td width="100" height="25" align="center" valign="middle" class="center">商品名称</td>
    <td width="100" height="25" align="center" valign="middle" class="center">购买数量</td>
    <td width="100" height="25" align="center" valign="middle" class="center">市场价格</td>
    <td width="100" height="25" align="center" valign="middle" class="center">会员价格</td>
    <td width="100" height="25" align="center" valign="middle" class="center">折扣率</td>
    <td width="100" height="25" align="center" valign="middle" class="right">合计</td>
  </tr>
{foreach key=key item=item from=$commarr}
  <tr>
    <td height="25" align="center" valign="middle" class="left"><input id="chk" name="chk[]" type="checkbox"
value="{$item.id}"></td>
    <td height="25" align="center" valign="middle" class="center"><div id = "c_name{$key}">  
{$item.name}</div></td>
    <td height="25" align="center" valign="middle" class="center"><input id="cnum{$key}" name="cnum{$key}"
type="text" class="shorttxt" value="{$item.num}" onkeyup="cvp({$key},{$item.v_price},{$shoparr})"></td>
    <td height="25" align="center" valign="middle" class="center"><div id="m_price{$key}"> {$item.m_price}
</div></td>
    <td height="25" align="center" valign="middle" class="center"><div id="v_price{$key}"> {$item.v_price}
</div></td>
    <td height="25" align="center" valign="middle" class="center"><div id="fold{$key}"> {$item.fold}
</div></td>
    <td height="25" align="center" valign="middle" class="right"><div id="total{$key}"> {$item.total}
</div></td>
  </tr>
{/foreach}
  <tr>
    <td height="25" colspan="3" align="left" valign="middle">
    <a href="#" onclick="return alldel(myshopcar)">全选</a> <a href="#" onclick="return overdel(myshopcar);">
反选</a>  
      <input type="button" value="删除选择" class="btn" style="border-color: #FFFFFF;" onClick = 'return
del(myshopcar);'>
        </td>
```

```
        <td height="25" align="center" valign="middle"><input id="cont" name="cont" type="button" class="btn" value="继续购物" onclick="return conshop(myshopcar)" /></td>
        <td height="25" align="center" valign="middle"><input id="uid" name="uid" type="hidden" value="{$smarty.session.member}" ><input id="settle" name="settle" type="button" class="btn" value="去收银台" onclick="return formset(form)" /></td>
        <td height="25" colspan="2" align="right" valign="middle"><div id='sum'>共计：{$sum} 元</div></td>
        </tr>
</form>
</table>
```

26.9.4 更改商品数量

对于新添加的商品，默认的购买数量为 1，在购物车页面可以对商品的数量进行修改。当商品数量发生变化时商品的“合计”金额和商品总金额会自动发生改变，该功能是通过触发 text 文本域的 onkeyup 事件调用 cvp()函数实现的。cvp()函数有 3 个参数，分别是商品 id、商品单价和商品类别。

首先，通过商品的 id 可以得到要修改商品的相关表单和标签属性。然后，通过商品单价和输入的商品数量计算该商品的合计金额。接着，使用 for 循环得到其他商品的合计金额。最后，将所有的合计金额累加，并输出到购物车页面。cvp()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
function cvp(key,vpr,shoparr){
        var n_pre = 'total';
        var num = 'cnum'+key.toString();
        var total = n_pre+key.toString();
        var t_number = document.getElementById(num).value;
        var ttl = t_number * vpr;
        document.getElementById(total).innerHTML = ttl;
        var sm = 0;

        for(var i = 0; i < shoparr; i++){

                var aaa = document.getElementById(n_pre+i.toString()).innerText;
                sm += parseInt(aaa);
        }
        document.getElementById('sum').innerHTML = '共计：'+sm+' 元';

}
```

这里所更改的商品数量，并没有被保存到数据库中，如果希望保存，那么单击“继续购物”按钮，则可以将商品数量更新到数据库中。

26.9.5 删除商品

当对添加的商品不满意时，可以对商品进行删除操作。操作流程为：首先选中要删除的商品前面的复选框，如果全部删除，则可以单击“全选”按钮或“反选”按钮。然后再单击“删除选择”按钮，在弹出的警告框中单击“确定”按钮，商品将被全部删除。删除商品的页面的运行结果如图 26.21 所示。

所有的删除操作都是通过 JS 脚本文件 shopcar.js 来实现的，相关的函数包括 alldel()、overdel()和 del()。

alldel()和 overdel()函数实现的原理比较简单，通过触发 onclick 事件来改变复选框的选中状态。函数代码如下：

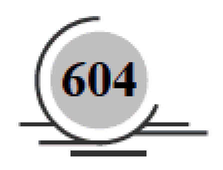




图 26.21　删除商品流程

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//全部选择/取消
function alldel(form){
	var leng = form.chk.length;
	if(leng==undefined){
	    if(!form.chk.checked)
		    form.chk.checked=true;
	 }else{
	   for( var i = 0; i < leng; i++)
	     {
		    if(!form.chk[i].checked)
			    form.chk[i].checked = true;
	     }
	 }
	return false;
}
//反选
function overdel(form){
	 var leng = form.chk.length;
	 if(leng==undefined){
	    if(!form.chk.checked)
		    form.chk.checked=true;
	      else
		    form.chk.checked=false;
	 }else{
	   for( var i = 0; i < leng; i++)
	     {
		    if(!form.chk[i].checked)
			    form.chk[i].checked = true;
		    else
			    form.chk[i].checked = false;
	     }
	 }
	return false;
}
```

使用 alldel()或 overdel()选中复选框后，即可调用 del()函数来实现删除功能。del()函数首先使用 for 循环，将被选中的复选框的 value 值取出并存成数组，然后根据数组生成 url，并使用 xmlhttp 对象调用这个 url，当处理完毕后，根据返回值弹出提示或刷新本页。该函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
/*  删除记录  */
function del(form){
	if(!window.confirm('是否要删除数据??')){

	}else{
		var leng = form.chk.length;
		if(leng==undefined){
			if(!form.chk.checked){
					alert('请选取要删除数据!');
			}else{
				rd = form.chk.value;
				var url = 'delshop.php?rd='+rd;
				xmlhttp.open("GET",url,true);
				xmlhttp.onreadystatechange = delnow;
				xmlhttp.send(null);
			}
		}else{
			var rd=new Array();
			var j = 0;
			for( var i = 0; i < leng; i++)
			{
				if(form.chk[i].checked){
					rd[j++] = form.chk[i].value;
				}
			}
			if(rd == ''){
				alert('请选取要删除数据!');
			}else{
				var url = "delshop.php?rd="+rd;
				xmlhttp.open("GET",url,true);
				xmlhttp.onreadystatechange = delnow;
				xmlhttp.send(null);
			}
		}
	}
	return false;
}
function delnow(){
	if(xmlhttp.readyState == 4){
		if(xmlhttp.status == 200){
			var msg = xmlhttp.responseText;
			if(msg != '1'){
				alert('删除失败'+msg);
			}else{
				alert('删除成功');
				location=('?page=shopcar');
			}
		}
	}
}
```

26.9.6　保存购物车

当用户希望保存商品更改后的商品数量时，可以单击“继续购物”按钮，将触发 onclick 事件调用 conshop()函数保存数据，该函数有一个参数，就是当前表单的名称。在 conshop()函数内，根据复选框和商品数量文



本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp   == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );              //设置文件编码格式
require("system/system.inc.php");                                 //包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp   == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
			if(msg == '1'){
				location='index.php';
			}else{
				alert('操作失败'+msg);
			}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");									//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");								//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");						//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp   == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
			if(msg == '1'){
				location='index.php';
			}else{
				alert('操作失败'+msg);
			}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );            //设置文件编码格式
require("system/system.inc.php");                               //包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp   == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );             //设置文件编码格式
require("system/system.inc.php");                                //包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");							//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
			if(msg == '1'){
				location='index.php';
			}else{
				alert('操作失败'+msg);
			}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");						//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
			if(msg == '1'){
				location='index.php';
			}else{
				alert('操作失败'+msg);
			}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );              //设置文件编码格式
require("system/system.inc.php");                                  //包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");							//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );		//设置文件编码格式
require("system/system.inc.php");				//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
			if(msg == '1'){
				location='index.php';
			}else{
				alert('操作失败'+msg);
			}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");						//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp	== '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );			//设置文件编码格式
require("system/system.inc.php");						//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp   == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
			if(msg == '1'){
				location='index.php';
			}else{
				alert('操作失败'+msg);
			}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );              //设置文件编码格式
require("system/system.inc.php");                                  //包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp   == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
			if(msg == '1'){
				location='index.php';
			}else{
				alert('操作失败'+msg);
			}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );				//设置文件编码格式
require("system/system.inc.php");						//包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

本域，生成两个数组 fst 和 snd，分别保存商品 id 和商品数量。

这里要注意，两个数组的值是要相互对应的，如商品 1 的 id 保存到 fst[1]中，那么商品 1 的数量就要保存到 snd[1]中，然后根据这两个数组生成一个 url，使用 xmlhttprequest 对象调用 url，最后根据回传信息作出相应的判断。conshop()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
//更改商品数量
function conshop(form){
	var n_pre = 'cnum';
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;
		var snd = form.cnum0.value;
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp    == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	var url = 'changecar.php?fst='+fst+'&snd='+snd;
	xmlhttp.open("GET",url,true);
	xmlhttp.onreadystatechange = updatecar;
	xmlhttp.send(null);
}
function updatecar(){
	if(xmlhttp.readyState == 4){
		var msg = xmlhttp.responseText;
		if(msg == '1'){
			location='index.php';
		}else{
			alert('操作失败'+msg);
		}
	}
}
```

在 conshop()函数中调用的 changecar.php 页为数据处理页，该页将商品 id 和商品数量进行重新排列，并保存到 shopping 字段内。该页面的代码如下：

（代码位置：光盘\TM\Instance\26\changecar.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
require("system/system.inc.php");                                   //包含配置文件
$sql = "select id,shopping from tb_user where name = '".$_SESSION['member']."'";
$rst = $admindb->ExecSQL($sql,$conn);
$reback = '0';
```

```
$changecar = array();
if(isset($_GET['fst']) && isset($_GET['snd'])){
	$fst = $_GET['fst'];
	$snd = $_GET['snd'];
	$farr = explode(',',$fst);
	$sarr = explode(',',$snd);
	$upcar = array();
	for($i = 0; $i < count($farr); $i++){
		$upcar[$i] = $farr[$i].','.$sarr[$i];
	}
	if(count($farr) > 1){
		$update = "update tb_user set shopping='".implode('@',$upcar)."' where name =
'".$_SESSION['member']."'";
	}else{
		$update = "update tb_user set shopping='".$upcar[0]."' where name = '".$_SESSION['member']."'";
	}
	$shop = $admindb->ExecSQL($update,$conn);
	if($shop){
		$reback = 1;
	}else{
		$reback = 2;
	}
}
echo $reback;
?>
```

26.10 收银台模块设计

视频讲解：光盘\TM\Video\第 26 章\收银台模块设计.exe

26.10.1 收银台模块概述

当用户停止浏览商品准备结账时，可以单击购物车页面中的"去收银台"按钮，该按钮将触发 onclick 事件调用 formset()函数显示订单页面，当用户提交订单后，系统将订单保存到数据表 tb_form 中，同时清空购物车，并显示订单信息提醒用户记录订单号。当货款发出后，还可以对订单进行查询。收银台页面的运行结果如图 26.22 所示。

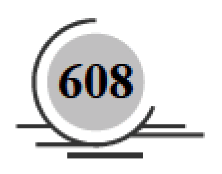
本节所涉及的页面有显示订单（formset()函数）、填写订单（settle.php、settle.tpl）、提交订单（settle_ chk.php）、反馈订单（forminfo.php、forminfo.tpl）和查询订单 5 部分。



图 26.22 收银台页面的运行结果

26.10.2 PDO 操作 MySQL 数据库技术

在收银台模块中，通过 PDO 中的方法完成订单信息的添加和数据的更新操作。同样调用数据库管理类 AdminDB 中的 ExecSQL()方法，完成数据的添加操作。



```
$changecar = array();
if(isset($_GET['fst']) && isset($_GET['snd'])){
	$fst = $_GET['fst'];
	$snd = $_GET['snd'];
	$farr = explode(',',$fst);
	$sarr = explode(',',$snd);
	$upcar = array();
	for($i = 0; $i < count($farr); $i++){
		$upcar[$i] = $farr[$i].','.$sarr[$i];
	}
	if(count($farr) > 1){
		$update = "update tb_user set shopping='".implode('@',$upcar)."' where name =
'".$_SESSION['member']."'";
	}else{
		$update = "update tb_user set shopping='".$upcar[0]."' where name = '".$_SESSION['member']."'";
	}
	$shop = $admindb->ExecSQL($update,$conn);
	if($shop){
		$reback = 1;
	}else{
		$reback = 2;
	}
}
echo $reback;
?>
```

26.10 收银台模块设计

视频讲解：光盘\TM\Video\第 26 章\收银台模块设计.exe

26.10.1 收银台模块概述

当用户停止浏览商品准备结账时，可以单击购物车页面中的“去收银台”按钮，该按钮将触发 onclick 事件调用 formset()函数显示订单页面，当用户提交订单后，系统将订单保存到数据表 tb_form 中，同时清空购物车，并显示订单信息提醒用户记录订单号。当货款发出后，还可以对订单进行查询。收银台页面的运行结果如图 26.22 所示。

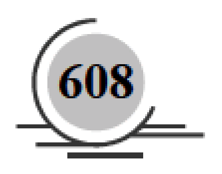
本节所涉及的页面有显示订单（formset()函数）、填写订单（settle.php、settle.tpl）、提交订单（settle_ chk.php）、反馈订单（forminfo.php、forminfo.tpl）和查询订单 5 部分。



图 26.22 收银台页面的运行结果

26.10.2 PDO 操作 MySQL 数据库技术

在收银台模块中，通过 PDO 中的方法完成订单信息的添加和数据的更新操作。同样调用数据库管理类 AdminDB 中的 ExecSQL()方法，完成数据的添加操作。



26.10.3　显示订单

订单信息提交页面的输出由 formset()函数决定，它将商品信息整理，通过 open 方法打开 settle.php 页来显示订单，并将整理后的商品信息传递到 settle.php 文件中。formset()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
function formset(form){
var uid = form.uid.value;
var n_pre = 'cnum';                                    //数量
    var lang = form.chk.length;
    if(lang == undefined){
        var fst = form.chk.value;                      //商品 id
        var snd = form.cnum0.value;                    //购买数量
    }else{
        var fst= new Array();
        var snd = new Array();
        for(var i = 0; i < lang; i++){
            var nm = n_pre+i.toString();
            var stmp = document.getElementById(nm).value;
            if(stmp    == '' || isNaN(stmp)){
                alert('不允许为空、必须为数字');
                document.getElementById(nm).select();
                return false;
            }
            snd[i] = stmp;
            var ftmp = form.chk[i].value;
            fst[i] = ftmp;
        }
    }
    open('settle.php?uid='+uid+'&fst='+fst+'&snd='+snd,'_blank','width=500 height=450',false);
}
```

说明　因为 open 方法使用了_blank 参数来打开一个新的页面，session 值传不过去，所以这里使用隐藏域来传递用户名称。

26.10.4　填写订单

settle.php 直接将接收的值传给 settle.tpl 模板，并载入 settle.tpl 模板。settle.php 页面的代码如下：

（代码位置：光盘\TM\Instance\26\settle.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );           //设置文件编码格式
require("system/system.inc.php");                              //包含配置文件
$fst = $_GET['fst'];
$snd = $_GET['snd'];
$uid = $_GET['uid'];
$smarty->assign('title','收银台');
$smarty->assign('fst',$fst);
$smarty->assign('snd',$snd);
$smarty->assign('uid',$uid);
```

26.10.3 显示订单

订单信息提交页面的输出由 formset()函数决定，它将商品信息整理，通过 open 方法打开 settle.php 页来显示订单，并将整理后的商品信息传递到 settle.php 文件中。formset()函数的代码如下：

（代码位置：光盘\TM\Instance\26\js\shopcar.js）

```
function formset(form){
var uid = form.uid.value;
var n_pre = 'cnum';                                    //数量
	var lang = form.chk.length;
	if(lang == undefined){
		var fst = form.chk.value;                      //商品 id
		var snd = form.cnum0.value;                    //购买数量
	}else{
		var fst= new Array();
		var snd = new Array();
		for(var i = 0; i < lang; i++){
			var nm = n_pre+i.toString();
			var stmp = document.getElementById(nm).value;
			if(stmp   == '' || isNaN(stmp)){
				alert('不允许为空、必须为数字');
				document.getElementById(nm).select();
				return false;
			}
			snd[i] = stmp;
			var ftmp = form.chk[i].value;
			fst[i] = ftmp;
		}
	}
	open('settle.php?uid='+uid+'&fst='+fst+'&snd='+snd,'_blank','width=500 height=450',false);
}
```

说明 因为 open 方法使用了_blank 参数来打开一个新的页面，session 值传不过去，所以这里使用隐藏域来传递用户名称。

26.10.4 填写订单

settle.php 直接将接收的值传给 settle.tpl 模板，并载入 settle.tpl 模板。settle.php 页面的代码如下：

（代码位置：光盘\TM\Instance\26\settle.php）

```
<?php
session_start();
header ( "Content-type: text/html; charset=UTF-8" );        //设置文件编码格式
require("system/system.inc.php");                           //包含配置文件
$fst = $_GET['fst'];
$snd = $_GET['snd'];
$uid = $_GET['uid'];
$smarty->assign('title','收银台');
$smarty->assign('fst',$fst);
$smarty->assign('snd',$snd);
$smarty->assign('uid',$uid);
```

```
$smarty->display('settle.tpl');
?>
```

settle.tpl 模板显示一个表单，这个表单的内容需要用户来填写，包括收货人、联系电话等信息。而从 PHP 页传过来的几个变量则被保存到隐藏域以传递到处理页，在表单中将数据提交到 settle_chk.php 处理页。

26.10.5　处理订单

处理页 settle_chk.php 获取表单中提交的数据，根据用户提交的商品信息，重新查找数据表 tb_commo，并从数据表中提取商品信息，保存到数组中，然后处理页将数组作为一条记录添加到表 tb_form 内。

数据添加成功的同时，处理页会根据 uid 找到该用户，将 shopping 字段清空，最后调用 forminfo.php 页来显示新添加的订单信息。settle_chk.php 页的代码如下：

（代码位置：光盘\TM\Instance\26\settle_chk.php）

```
<?php
header ( "Content-type: text/html; charset=UTF-8" );                //设置文件编码格式
require("system/system.inc.php");                                   //包含配置文件
$sql="insert  into  tb_form(formid,commo_id,commo_name,commo_num,agoprice,fold,total,vendee,taker,address,
tel,code, pay_method,del_method,formtime,state)values(";
$formid=time();
$tmpid = explode(',',$_POST['fst']);
$tmpnm = explode(',',$_POST['snd']);
$number = count($tmpid);
$tmpna = array();
$tmpvp = array();
$tmpfd = array();
$tmptt = 0;
if($number >1){
	for($i = 0; $i < $number; $i++){
		$tmpsql = "select name,v_price,fold from tb_commo where id = '".$tmpid[$i]."'";
		$tmprst = $admindb->ExecSQL($tmpsql,$conn);
		$tmpna[$i] = $tmprst[0]['name'];
		$tmpvp[$i] = $tmprst[0]['v_price'];
		$tmpfd[$i] = $tmprst[0]['fold'];
		$tmptt += $tmprst[0]['v_price'] * $tmpnm[$i];
		@$tmpsell = $tmprst[0]['sell'] + 1;
		$addsql = "update tb_commo set sell = '".$tmpsell."' where id = '".$tmpid[$i]."'";
		$addrst = $admindb->ExecSQL($addsql,$conn);
	}
	$sql.="'".$formid."','".$_POST['fst']."','".implode(',',$tmpna)."','".$_POST['snd']."','".implode(',',$tmpvp)."','".im
plode(',',$tmpfd)."','".$tmptt."','".$_POST['uid']."'";
}else if($number == 1){
	$tmpsql = "select name,v_price,fold from tb_commo where id = '".$tmpid[0]."'";
	$tmprst = $admindb->ExecSQL($tmpsql,$conn);
	$tmptt= $tmprst[0]['v_price'] * $tmpnm[0];
	@$tmpsell = $tmprst[0]['sell'] + 1;
	$addsql = "update tb_commo set sell = '".$tmpsell."' where id = '".$tmpid[0]."'";
	$addrst = $admindb->ExecSQL($addsql,$conn);
	$sql.="'".$formid."','".$_POST['fst']."','".$tmprst[0]['name']."','".$_POST['snd']."','".$tmprst[0]['v_price']."','".$t
mprst[0]['fold']."','".$tmptt."','".$_POST['uid']."'";
}else{
	echo 'error';
	exit();
}
$sql.=",'".$_POST['taker']."','".$_POST['address']."','".$_POST['tel']."','".$_POST['code']."','".$_POST['pay']."','".$
_POST['del']."','".date("Y-m-d H:i:s")."',0)";
```

```
$InsertSQL = $admindb->ExecSQL($sql,$conn);
if(false == $InsertSQL){
	echo "<script>alert('购买失败');history.back;</script>";
}else{
	$updsql = "update tb_user set consume='".$tmptt."',shopping='' where name = '".$_POST['uid']."'";
	$updrst = $admindb->ExecSQL($updsql,$conn);
	echo "<script>top.opener.location.reload();</script>";
	echo "<script>open('forminfo.php?fid=$formid','_blank','width=750 height=650',false);window.close();</script>";
}
?>
```

由于篇幅所限，有关反馈订单和查询订单的内容这里不再讲解，请读者参考本书光盘中的源代码。

26.11　后台首页设计

视频讲解：光盘\TM\Video\第 26 章\后台首页设计.exe

26.11.1　后台首页概述

后台管理系统是网站管理员对商品、会员及公告等信息进行统一管理的场所，本系统的后台主要包括以下功能。

- ☑　类别管理模块：主要包括对商品类别的添加、修改及删除操作。
- ☑　商品管理模块：主要包括对商品的添加、修改、删除及订单处理。
- ☑　用户管理模块：主要包括管理员管理和会员管理。其中管理员管理是实现对管理员的添加、删除和修改功能，会员管理则包括删除和冻结功能。
- ☑　公告管理模块：主要包括公告的添加及删除操作。
- ☑　链接管理模块：主要包括添加、修改和删除友情链接。

后台首页的运行结果如图 26.23 所示。

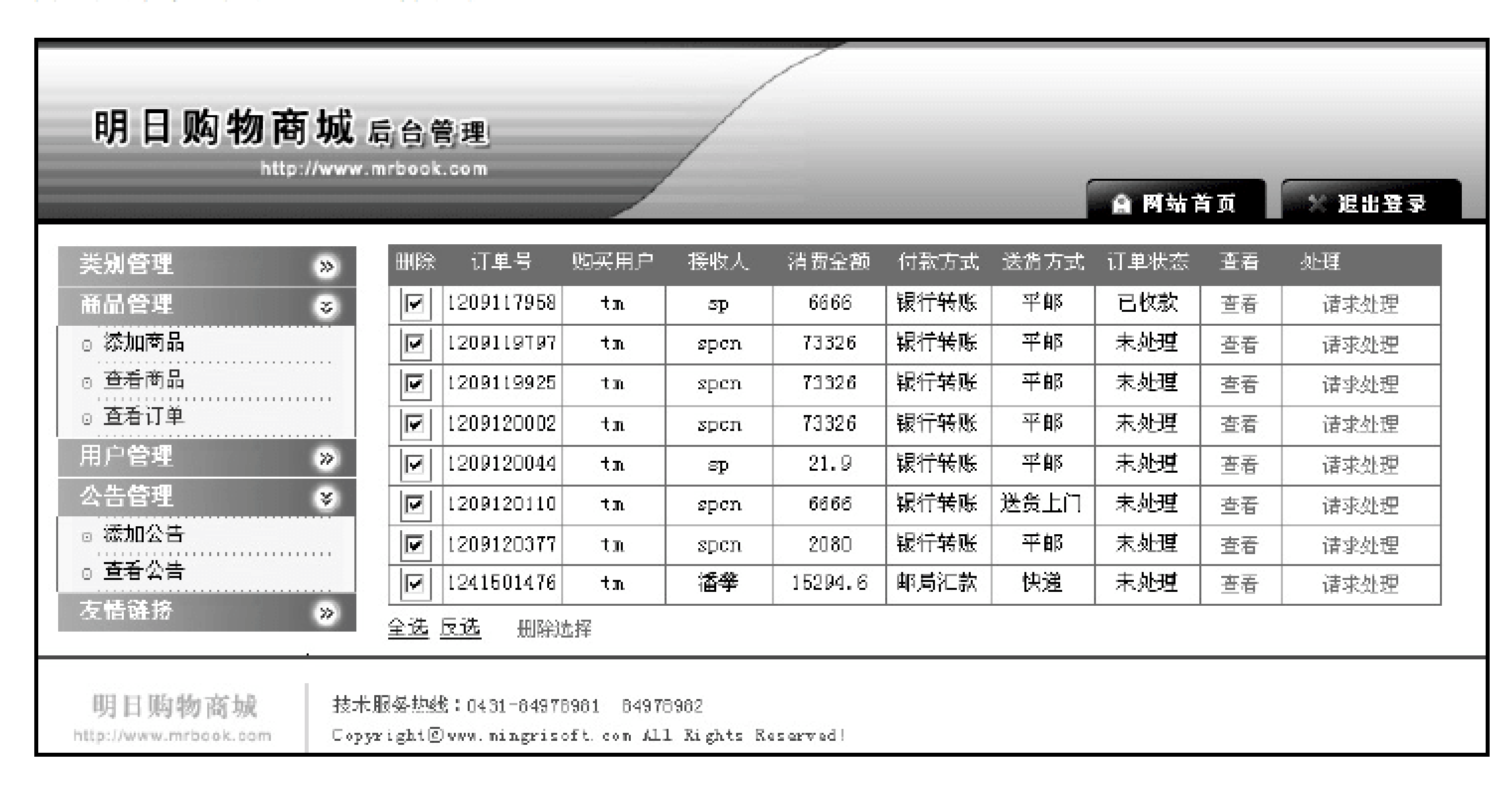


图 26.23　后台首页运行的结果

26.11.2　后台网页布局技术

后台首页和前台首页不同，其使用的是框架布局。框架布局的特点是：可以将容器窗口划分为若干个

```
$InsertSQL = $admindb->ExecSQL($sql,$conn);
if(false == $InsertSQL){
	echo "<script>alert('购买失败');history.back;</script>";
}else{
	$updsql = "update tb_user set consume='".$tmptt."',shopping='' where name = '".$_POST['uid']."'";
	$updrst = $admindb->ExecSQL($updsql,$conn);
	echo "<script>top.opener.location.reload();</script>";
	echo "<script>open('forminfo.php?fid=$formid','_blank','width=750 height=650',false);window.close();</script>";
}
?>
```

由于篇幅所限，有关反馈订单和查询订单的内容这里不再讲解，请读者参考本书光盘中的源代码。

26.11　后台首页设计

视频讲解：光盘\TM\Video\第 26 章\后台首页设计.exe

26.11.1　后台首页概述

后台管理系统是网站管理员对商品、会员及公告等信息进行统一管理的场所，本系统的后台主要包括以下功能。

- ☑ 类别管理模块：主要包括对商品类别的添加、修改及删除操作。
- ☑ 商品管理模块：主要包括对商品的添加、修改、删除及订单处理。
- ☑ 用户管理模块：主要包括管理员管理和会员管理。其中管理员管理是实现对管理员的添加、删除和修改功能，会员管理则包括删除和冻结功能。
- ☑ 公告管理模块：主要包括公告的添加及删除操作。
- ☑ 链接管理模块：主要包括添加、修改和删除友情链接。

后台首页的运行结果如图 26.23 所示。

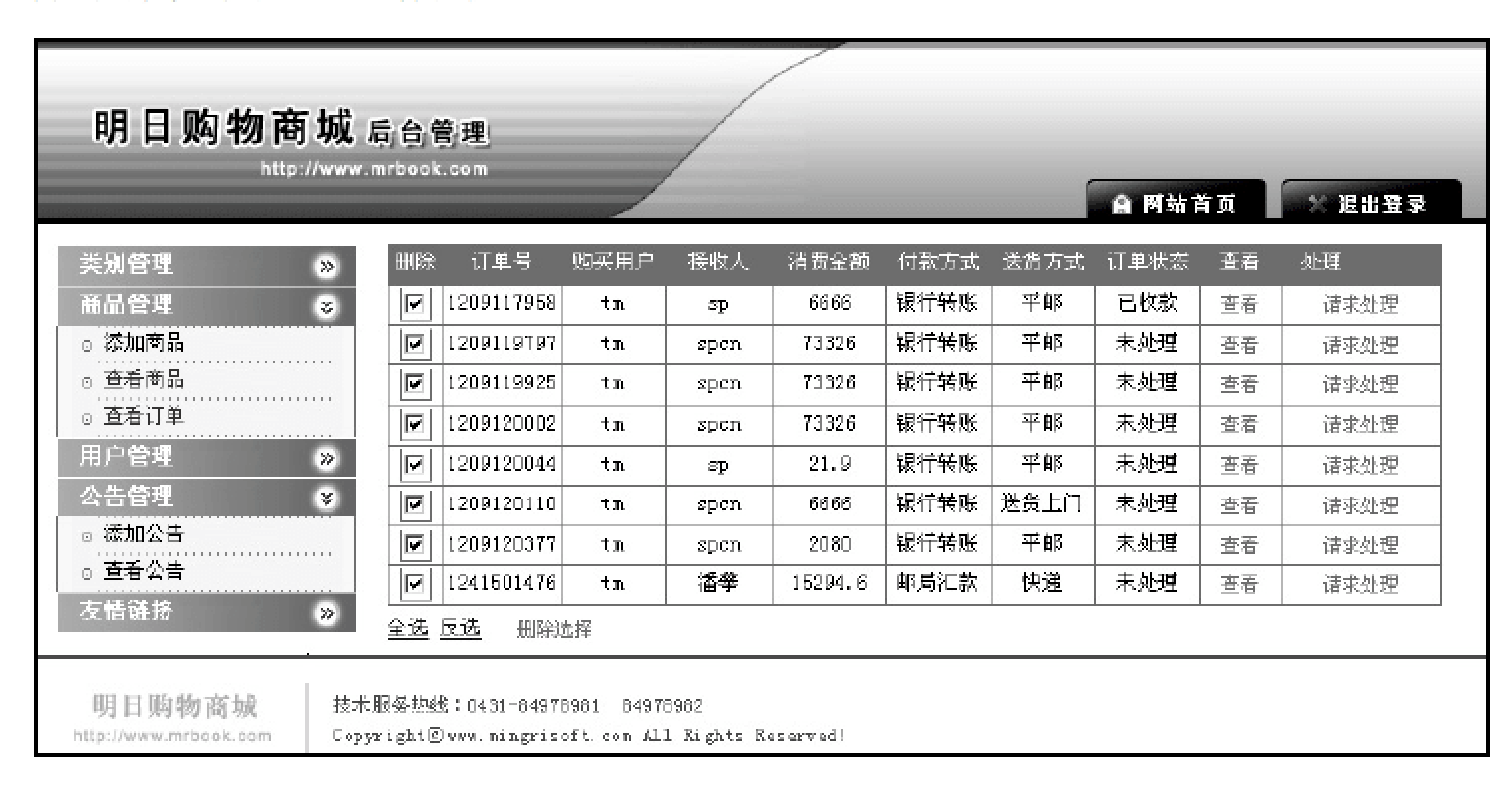


图 26.23　后台首页运行的结果

26.11.2　后台网页布局技术

后台首页和前台首页不同，其使用的是框架布局。框架布局的特点是：可以将容器窗口划分为若干个

子窗口，每个子窗口可以分别显示不同的网页，网页之间为相互独立的，没有直接的关联，又由一个网页将这些分开的网页组成一个完整的网页，显示在浏览者的浏览器中。框架布局的好处是：每次浏览者发出对页面的请求时，只下载发生变化的框架页面，其他子页面保持不变。下面来具体看一下框架布局的使用格式及属性。

1．框架布局格式

框架布局的格式很简单，只要几行代码即可。常用的格式如下：

```
<html>
<head>
…
</head>
<frameset>
    <frame>
    <frame>
</frameset>
<noframes>
     <body>
     …
     </body>
</noframes>
</html>
```

其中<frameset>和<frame>标签是框架集标记，而<noframes>标签是为了防止浏览器不支持框架而实行的一种补救措施。如果浏览器不支持框架集，就会执行<noframes>标签里的内容，让用户能够正常浏览网页。

2．框架集属性

框架集包含各个框架的信息，通过<frameset>标签来定义。框架是按照行和列来组织的，可以使用<frameset>标签的属性对框架的结构进行设置。下面给出框架集的常用属性值、说明和应用举例，如表 26.1 所示。

表 26.1　框架集的常用属性

参　　数	说　　明	描　　述
COLS	在水平方向上将浏览器分割成多个窗口，取值有 3 种形式：像素、百分比（%）和相对尺寸（*）	<frameset cols="25%,100,*" > <frame></frame> </frameset>
ROWS	在垂直方向上将浏览器分割成多个窗口，取值和 COLS 类似，也是 3 种形式	<frameset rows="25%,100,*" > <frame> <frame> </frameset>
FRAMEBORDER	指定框架周围是否显示边框，取值为 1（显示边框，默认值）或 0（不显示边框）	<framset cols="25%,*" cols="*" frameborder="0"> … </frameset>
FRAMESPACING	指定框架之间的间隔，以像素为单位。默认是无间隔的	<framset cols="25%,*" cols="*" framespacing="1"> … </frameset>
BORDER	指定边框的宽度，frameborder 属性为 1 时该属性才有效	<framset cols="25%,*" cols="*" frameborder="1" border="5"> … </frameset>

子窗口，每个子窗口可以分别显示不同的网页，网页之间为相互独立的，没有直接的关联，又由一个网页将这些分开的网页组成一个完整的网页，显示在浏览者的浏览器中。框架布局的好处是：每次浏览者发出对页面的请求时，只下载发生变化的框架页面，其他子页面保持不变。下面来具体看一下框架布局的使用格式及属性。

1. 框架布局格式

框架布局的格式很简单，只要几行代码即可。常用的格式如下：

```
<html>
<head>
…
</head>
<frameset>
    <frame>
    <frame>
</frameset>
<noframes>
     <body>
     …
     </body>
</noframes>
</html>
```

其中<frameset>和<frame>标签是框架集标记，而<noframes>标签是为了防止浏览器不支持框架而实行的一种补救措施。如果浏览器不支持框架集，就会执行<noframes>标签里的内容，让用户能够正常浏览网页。

2. 框架集属性

框架集包含各个框架的信息，通过<frameset>标签来定义。框架是按照行和列来组织的，可以使用<frameset>标签的属性对框架的结构进行设置。下面给出框架集的常用属性值、说明和应用举例，如表 26.1 所示。

表 26.1　框架集的常用属性

参　　数	说　　明	描　　述
COLS	在水平方向上将浏览器分割成多个窗口，取值有 3 种形式：像素、百分比（%）和相对尺寸（*）	<frameset cols="25%,100,*" > <frame></frame> </frameset>
ROWS	在垂直方向上将浏览器分割成多个窗口，取值和 COLS 类似，也是 3 种形式	<frameset rows="25%,100,*" > <frame> <frame> </frameset>
FRAMEBORDER	指定框架周围是否显示边框，取值为 1（显示边框，默认值）或 0（不显示边框）	<framset cols="25%,*" cols="*" frameborder="0"> … </frameset>
FRAMESPACING	指定框架之间的间隔，以像素为单位。默认是无间隔的	<framset cols="25%,*" cols="*" framespacing="1"> … </frameset>
BORDER	指定边框的宽度，frameborder 属性为 1 时该属性才有效	<framset cols="25%,*" cols="*" frameborder="1" border="5"> … </frameset>

3．框架属性

使用<frame>标签可以设置框架的属性，包括框架的名称，框架是否包含滚动条以及在框架中显示的网页等。<frame>标签的常用属性及其说明如表 26.2 所示。

表 26.2 框架属性

参　数	说　明
NAME	指定框架的名称
SRC	指定在框架中显示的网页文件（包括 HMRL、PHP、JSP 等网页文件）
FRAMEBODER	指定框架周围是否显示边框，取值为 1（显示边框，为默认）或 0（不显示边框）
NORESIZE	可选属性，若指定了该属性，则不能调整框架的大小
SCROLLING	指定框架是否包含滚动条。属性可以是 yes（有）、no（没有）和 auto（自由）

26.11.3 后台首页实现过程

（1）定义框架页面 main.tpl，包含 3 个文件：top.tpl、left.php 和 default.php。main.tpl 页的代码如下：

（代码位置：光盘\TM\Instance\26\admin\system\templates\main.tpl）

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>明日购物商城后台管理系统</title>
<link rel="stylesheet" href="css/style.css" />
</head>
<frameset rows="113,*,100" cols="1004" frameborder="no" border="0" framespacing="0">
  <frame src="top.php" name="topFrame" scrolling="No" noresize="noresize" id="topFrame" title="topFrame" />
  <frameset rows="*" cols="10%,210,*,10%" framespacing="0" frameborder="no" border="0">

<frame src="s.php" name="lFrame" frameborder="0" scrolling="auto" noresize="noresize" id="lFrame" title="leftFrame" />
    <frame src="left.php" name="leftFrame" frameborder="0" scrolling="auto" noresize="noresize" id="leftFrame" title="leftFrame" />
    <frame src="default.php" name="mainFrame" id="mainFrame" title="mainFrame" />
<frame src="s.php" name="rFrame" frameborder="0" scrolling="auto" noresize="noresize" id="rFrame" title="leftFrame" />
  </frameset>
   <frame src="bottom.php" name="bottomFrame" scrolling="No" noresize="noresize" id="bottomFrame" title="bottomFrame" />
</frameset>
<noframes><body>
</body>
</noframes>
</html>
```

（2）left.php 页是一个树形菜单，应用 DIV+JavaScript+CSS 实现。首先介绍 div 标签，在 left.tpl 模板文件中。其关键代码如下：

（代码位置：光盘\TM\Instance\26\admin\system\templates\left.tpl）

```
<!--  载入 CSS 样式和 JavaScript 脚本  -->
<link href="css/left.css" rel="stylesheet" type="text/css" />
<script language="javascript" src="js/left.js"></script>
<!--  类别管理菜单，注意加粗的地方  -->
```

```
<div id="type" align="center" onclick="javascript:change(one,type);">类别管理</div>
<!--  子菜单  -->
<div id="one" style="display: ">
<div id="addtype" align="center"><a href="addtype.php" target="mainFrame" id="menu">添加类别</a></div>
<div id="showtype" align="center"><a href="showtype.php" target="mainFrame" id="menu">查看类别</a></div>
</div>
<div id="hidediv" align="center"></div>
<!--  商品管理菜单  -->
<div id="commo" align="center" onclick="javascript:change(two,type);">类别管理</div>
<div id="two"style="display:none">
<!--  商品管理子菜单  -->
…
</div>
…
```

说明 除了加粗的 id 名称和 js 事件不同外，其他菜单的结构完全相同，此时只需修改超链接即可。

注意 除了第一个类别菜单的子菜单 display 样式为空外，其他几个子菜单的 div 样式都为“display=none;”。

该页面在 Dreamweaver 中的效果如图 26.24 所示。

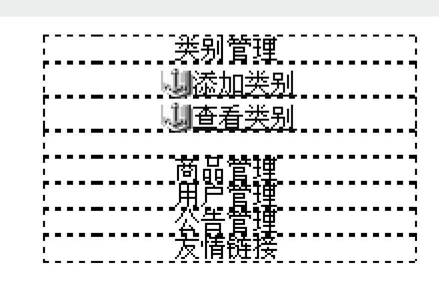


图 26.24　div 树形菜单

因为其他子菜单的样式为 display=none，所以只有“类别管理”子菜单是可见的，下面为它添加 js 事件。left.js 脚本文件的代码如下：

（代码位置：光盘\TM\Instance\26\admin\js\left.js）

```
function change(nu,lx){
	if(nu.style.display == "none"){
		nu.style.display = "";
		lx.style.background="url(images/admin(5).gif)";
	}else{
		nu.style.display = "none";
		lx.style.background="url(images/admin(1).gif)";
	}
}
```

最后在 left.css 中设置 div 的长、宽等一些默认参数。一个简单而又实用的树形菜单就完成了。对于后台的大部分模块来说，其功能实现的方法和开发步骤在前台的模块设计中基本都已经介绍过。由于篇幅所限，这里不再对后台管理模块进行详细讲解。

26.12　小　　结

本项目使用 Smarty、PDO 数据库抽象层和 Ajax 等目前的主流技术，开发一个电子商务平台。通过本章的学习，读者不仅可以了解电子商务网站的开发流程，而且还可以熟悉购物车、订单及加密技术的开发思想。

```
<div id="type" align="center" onclick="javascript:change(one,type);">类别管理</div>
<!--  子菜单  -->
<div id="one" style="display: ">
<div id="addtype" align="center"><a href="addtype.php" target="mainFrame" id="menu">添加类别</a></div>
<div id="showtype" align="center"><a href="showtype.php" target="mainFrame" id="menu">查看类别</a></div>
</div>
<div id="hidediv" align="center"></div>
<!--  商品管理菜单  -->
<div id="commo" align="center" onclick="javascript:change(two,type);">类别管理</div>
<div id="two"style="display:none">
<!--  商品管理子菜单  -->
…
</div>
…
```

说明 除了加粗的 id 名称和 js 事件不同外，其他菜单的结构完全相同，此时只需修改超链接即可。

注意 除了第一个类别菜单的子菜单 display 样式为空外，其他几个子菜单的 div 样式都为“display=none;”。

该页面在 Dreamweaver 中的效果如图 26.24 所示。

因为其他子菜单的样式为 display=none，所以只有“类别管理”子菜单是可见的，下面为它添加 js 事件。left.js 脚本文件的代码如下：

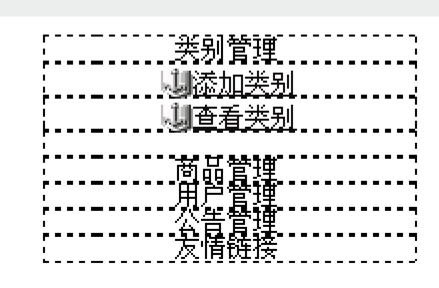


图 26.24　div 树形菜单

（代码位置：光盘\TM\Instance\26\admin\js\left.js）

```
function change(nu,lx){
	if(nu.style.display == "none"){
		nu.style.display = "";
		lx.style.background="url(images/admin(5).gif)";
	}else{
		nu.style.display = "none";
		lx.style.background="url(images/admin(1).gif)";
	}
}
```

最后在 left.css 中设置 div 的长、宽等一些默认参数。一个简单而又实用的树形菜单就完成了。对于后台的大部分模块来说，其功能实现的方法和开发步骤在前台的模块设计中基本都已经介绍过。由于篇幅所限，这里不再对后台管理模块进行详细讲解。

26.12　小　　结

本项目使用 Smarty、PDO 数据库抽象层和 Ajax 等目前的主流技术，开发一个电子商务平台。通过本章的学习，读者不仅可以了解电子商务网站的开发流程，而且还可以熟悉购物车、订单及加密技术的开发思想。

第 6 篇

项目实战

第27章

易查供求信息网

（视频讲解：125 分钟）

在全球知识经济和信息化高速发展的今天，信息化是决定企业成败的关键因素，企业需要在网站上发布供求信息，以促使企业在同领域中得到突飞猛进的发展。

一个广泛的、快速的、自由的信息交流平台，为用户带来方便的同时，也会给企业带来无限商机。于是，以因特网为基础的信息交流平台即易查供求信息网出现了。易查供求信息网致力于优化信息交流，实现信息的快速交流。

通过阅读本章内容，你可以：

- 了解使当前窗口承载框架页中的超链接页面
- 了解如何自动计算以系统日期为基数的相对日期
- 了解 do…while 循环语句的应用
- 了解查询关键字描红技术
- 了解框架技术在 Web 网站中的应用
- 了解表单数据的两种提交方式

第27章

易查供求信息网

（视频讲解：125 分钟）

在全球知识经济和信息化高速发展的今天，信息化是决定企业成败的关键因素，企业需要在网站上发布供求信息，以促使企业在同领域中得到突飞猛进的发展。

一个广泛的、快速的、自由的信息交流平台，为用户带来方便的同时，也会给企业带来无限商机。于是，以因特网为基础的信息交流平台即易查供求信息网出现了。易查供求信息网致力于优化信息交流，实现信息的快速交流。

通过阅读本章内容，你可以：

- 了解使当前窗口承载框架页中的超链接页面
- 了解如何自动计算以系统日期为基数的相对日期
- 了解 do…while 循环语句的应用
- 了解查询关键字描红技术
- 了解框架技术在 Web 网站中的应用
- 了解表单数据的两种提交方式

27.1　易查供求信息网概述

吉林省明日科技有限公司是一家以整合渠道资源为主的高科技公司。为了扩大企业规模，增强企业的竞争力，该公司决定向多元化发展，借助 Internet 在国内的快速发展，聚集部分资金投入网站建设，为企业和用户提供综合信息服务，以向企业提供有偿信息服务为盈利方式，打造一个全新的供求信息网。例如，提供企业广告、发布各类免费供求信息、发布企业付费信息等服务方式。现需要委托其他单位开发一个综合信息网站。

27.2　系 统 分 析

27.2.1　需求分析

对于信息网站来说，用户的访问量是至关重要的。如果网站的访问量很低，那么就很少有企业会要求为他提供有偿服务，也就没有利润可言了。因此信息网站必须为用户提供大量的、免费的、有价值的信息才能够吸引用户。为此，网站不仅要为企业提供各种有偿服务，还需要额外为用户提供大量的无偿服务。通过与企业的实际接触和沟通，确定网站应包括招聘信息、求职信息、培训信息、公寓信息、家教信息、车辆信息、物品求购、物品出售、求兑出兑、寻求合作、企业广告等服务。

通过实际调查，要求供求信息网应具有以下功能。

☑　界面设计美观大方、方便、快捷、操作灵活，树立企业形象。
☑　实现强大的供求信息查询，支持模糊查询。
☑　用户不需要注册，便可免费发布供求信息。
☑　免费发布的供求信息必须经后台审核后才能正式发布，避免不良信息。
☑　支持海量数据录入。
☑　由于供求信息数据量大，后台应该可以随时清理数据。

27.2.2　可行性分析

根据《GB8567—88 计算机软件产品开发文件编制指南》中可行性分析的要求，制定可行性研究报告如下。

1. 引言

（1）编写目的

为了给企业的决策层提供是否进行项目实施的参考依据，现以文件的形式分析项目的风险、项目需要的投资与效益。

（2）背景

吉林省明日科技有限公司是一家以整合渠道资源为主的高科技公司。企业为了不断满足客户的需求，为达到企业在同行业领域中的领先地位，现需要委托其他公司开发一个综合信息网，项目名称为易查供求信息网。

27.1　易查供求信息网概述

吉林省明日科技有限公司是一家以整合渠道资源为主的高科技公司。为了扩大企业规模，增强企业的竞争力，该公司决定向多元化发展，借助 Internet 在国内的快速发展，聚集部分资金投入网站建设，为企业和用户提供综合信息服务，以向企业提供有偿信息服务为盈利方式，打造一个全新的供求信息网。例如，提供企业广告、发布各类免费供求信息、发布企业付费信息等服务方式。现需要委托其他单位开发一个综合信息网站。

27.2　系 统 分 析

27.2.1　需求分析

对于信息网站来说，用户的访问量是至关重要的。如果网站的访问量很低，那么就很少有企业会要求为他提供有偿服务，也就没有利润可言了。因此信息网站必须为用户提供大量的、免费的、有价值的信息才能够吸引用户。为此，网站不仅要为企业提供各种有偿服务，还需要额外为用户提供大量的无偿服务。通过与企业的实际接触和沟通，确定网站应包括招聘信息、求职信息、培训信息、公寓信息、家教信息、车辆信息、物品求购、物品出售、求兑出兑、寻求合作、企业广告等服务。

通过实际调查，要求供求信息网应具有以下功能。

☑　界面设计美观大方、方便、快捷、操作灵活，树立企业形象。
☑　实现强大的供求信息查询，支持模糊查询。
☑　用户不需要注册，便可免费发布供求信息。
☑　免费发布的供求信息必须经后台审核后才能正式发布，避免不良信息。
☑　支持海量数据录入。
☑　由于供求信息数据量大，后台应该可以随时清理数据。

27.2.2　可行性分析

根据《GB8567—88 计算机软件产品开发文件编制指南》中可行性分析的要求，制定可行性研究报告如下。

1. 引言

（1）编写目的

为了给企业的决策层提供是否进行项目实施的参考依据，现以文件的形式分析项目的风险、项目需要的投资与效益。

（2）背景

吉林省明日科技有限公司是一家以整合渠道资源为主的高科技公司。企业为了不断满足客户的需求，为达到企业在同行业领域中的领先地位，现需要委托其他公司开发一个综合信息网，项目名称为易查供求信息网。

2．可行性研究的前提

（1）要求

易查供求信息网要求能够提供信息搜索信息定位描红，发布免费信息，发布付费信息，发布企业广告，对各类发布的信息进行审核、删除、检索等功能。

（2）目标

易查供求信息网的主要目标是提供强大的搜索功能，准确的信息描红定位功能，付费信息的管理、免费信息的审核和删除功能。

（3）条件、假定和限制

项目需要在两个月内交付用户使用。系统分析师需要 3 天内到位，用户需要 4 天时间确认需求分析文档。去除员工两个月的正常休息日，程序开发人员需要在 1 个月零几天的时间内进行系统设计、程序编码、系统测试、程序调试和网站部署工作。

（4）评价尺度

根据用户的要求，系统应以搜索引擎为主，对于发布的供求信息应能及时准确地保存、审核、查询、描红定位。由于用户存在多个营业点，系统应具有局域网操作的能力，在多个营业点同时运行系统时，系统中各项操作的延时不能超过 10 秒钟。此外，在系统出现故障时，应能及时进行恢复。

3．投资及效益分析

（1）支出

根据系统的规模及两个月的项目开发周期，公司决定投入 5 个人。因此，公司将直接支付 8 万元的工资及各种福利待遇。在项目安装及调试阶段，用户培训、员工出差等费用支出需要 2 万元。在项目维护阶段预计需要投入 2 万元的资金，累计项目投入需要 12 万元资金。

（2）收益

用户提供项目资金 30 万元。对于项目运行后进行的改动，采取协商的原则根据改动规模额外提供资金。因此从投资与收益的效益比上，公司可以获得 18 万元的利润。

项目完成后，将给公司提供资源储备，包括技术、经验的积累，以后再开发类似的项目时，可以极大地缩短项目开发周期。

4．结论

根据上面的分析，技术上不会存在问题，因此项目延期的可能性很小。在效益上，公司投入 5 个人、两个月的时间获利 18 万元，比较可观。在公司今后的发展上可以储备网站开发的经验和资源，因此认为该项目可以开发。

27.3 系统设计

27.3.1 系统目标

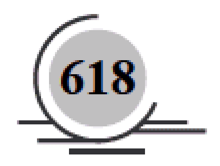
根据需求分析的描述以及与用户的沟通，现制定网站实现的目标如下。

☑ 系统采用人机对话方式，界面美观友好，界面简洁、框架清晰、美观大方。

☑ 灵活快速地填写供求信息，使信息传递更快捷。

☑ 信息查询灵活、方便，数据存储安全可靠。

☑ 实施强大的后台审核功能。



☑ 实现强大的搜索引擎，支持模糊查询、关键字描红功能等。
☑ 对用户输入的数据，系统进行严格的数据检验，尽可能排除人为的错误。
☑ 网站最大限度地实现易维护性和易操作性。
☑ 为充分展现网站的交互性，供求信息网采用动态网页技术实现用户信息在线发布。
☑ 具备完善的后台管理功能，能够及时、准确地对网站进行维护和更新。

27.3.2　系统功能结构

易查供求信息网前台功能结构图如图 27.1 所示。

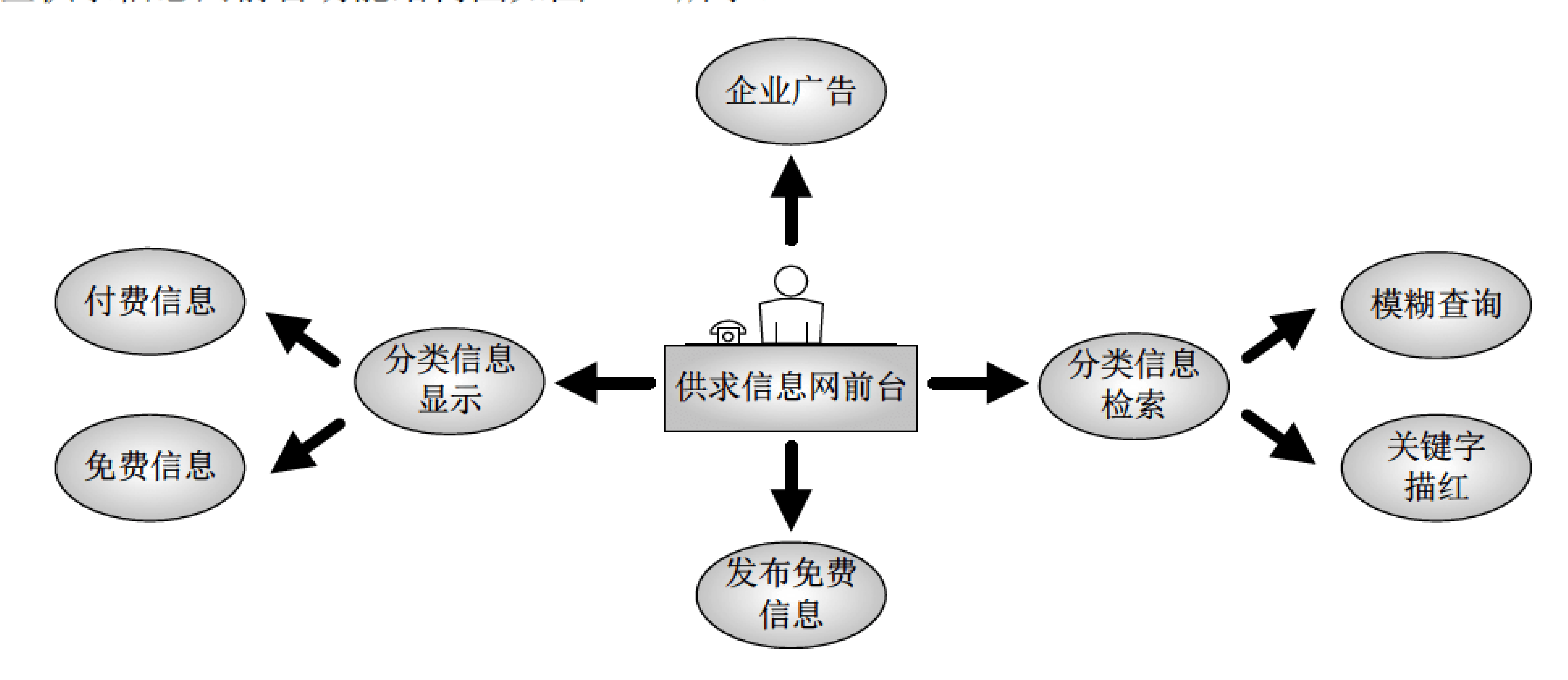


图 27.1　易查供求信息网前台功能结构图

易查供求信息网后台功能结构图如图 27.2 所示。

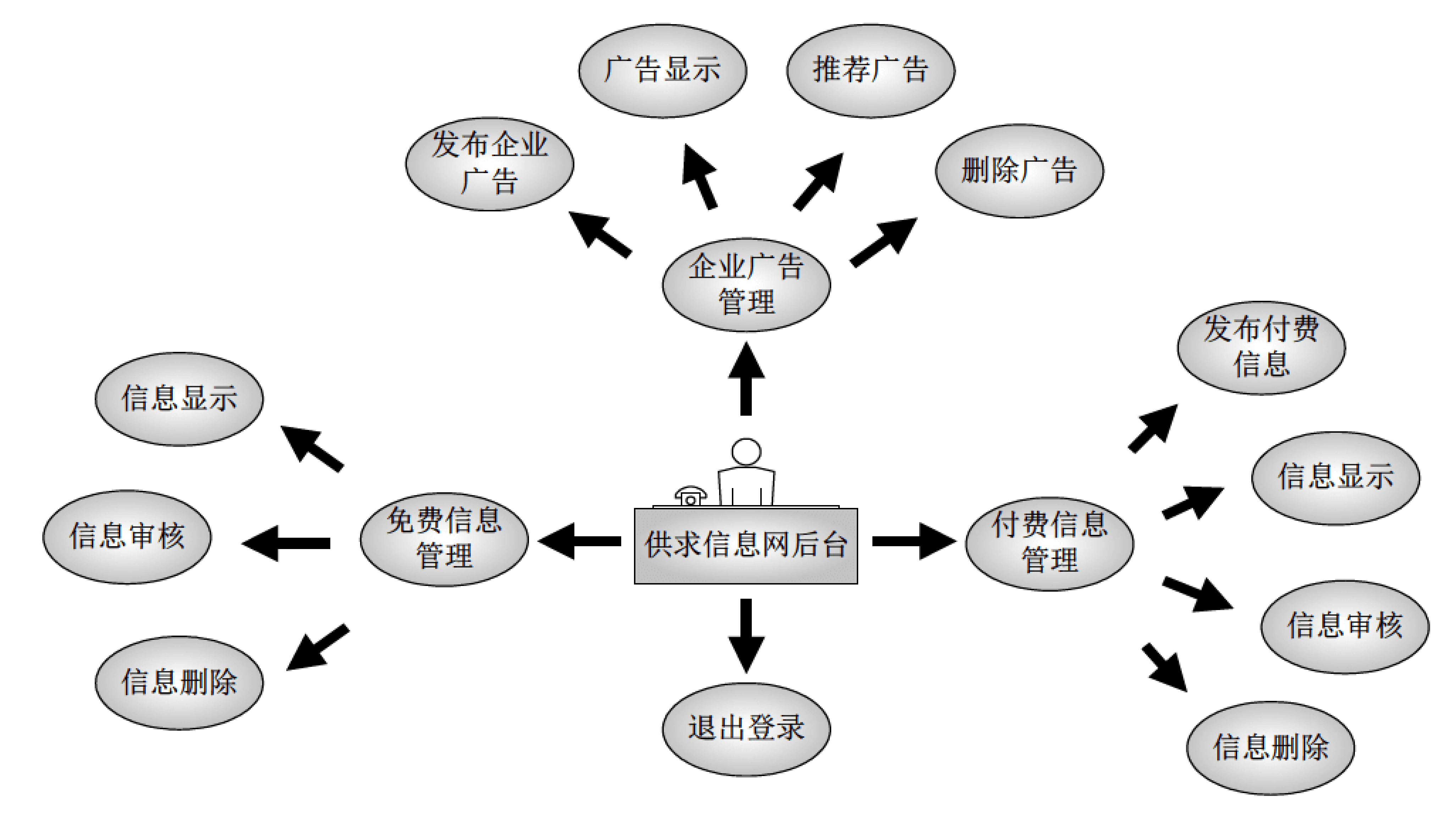


图 27.2　易查供求信息网后台功能结构图

易查供求信息网的系统流程如图 27.3 所示。

- ☑ 实现强大的搜索引擎，支持模糊查询、关键字描红功能等。
- ☑ 对用户输入的数据，系统进行严格的数据检验，尽可能排除人为的错误。
- ☑ 网站最大限度地实现易维护性和易操作性。
- ☑ 为充分展现网站的交互性，供求信息网采用动态网页技术实现用户信息在线发布。
- ☑ 具备完善的后台管理功能，能够及时、准确地对网站进行维护和更新。

27.3.2　系统功能结构

易查供求信息网前台功能结构图如图 27.1 所示。

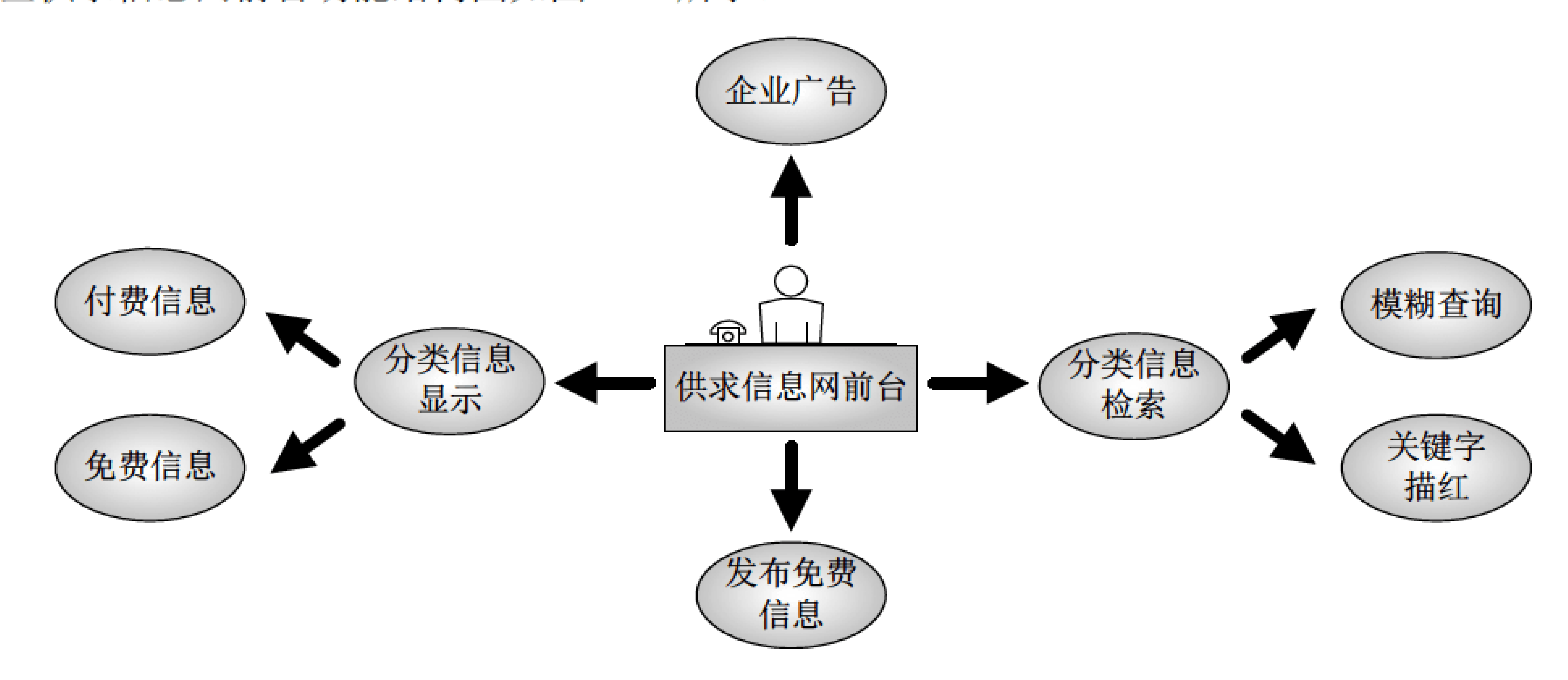


图 27.1　易查供求信息网前台功能结构图

易查供求信息网后台功能结构图如图 27.2 所示。

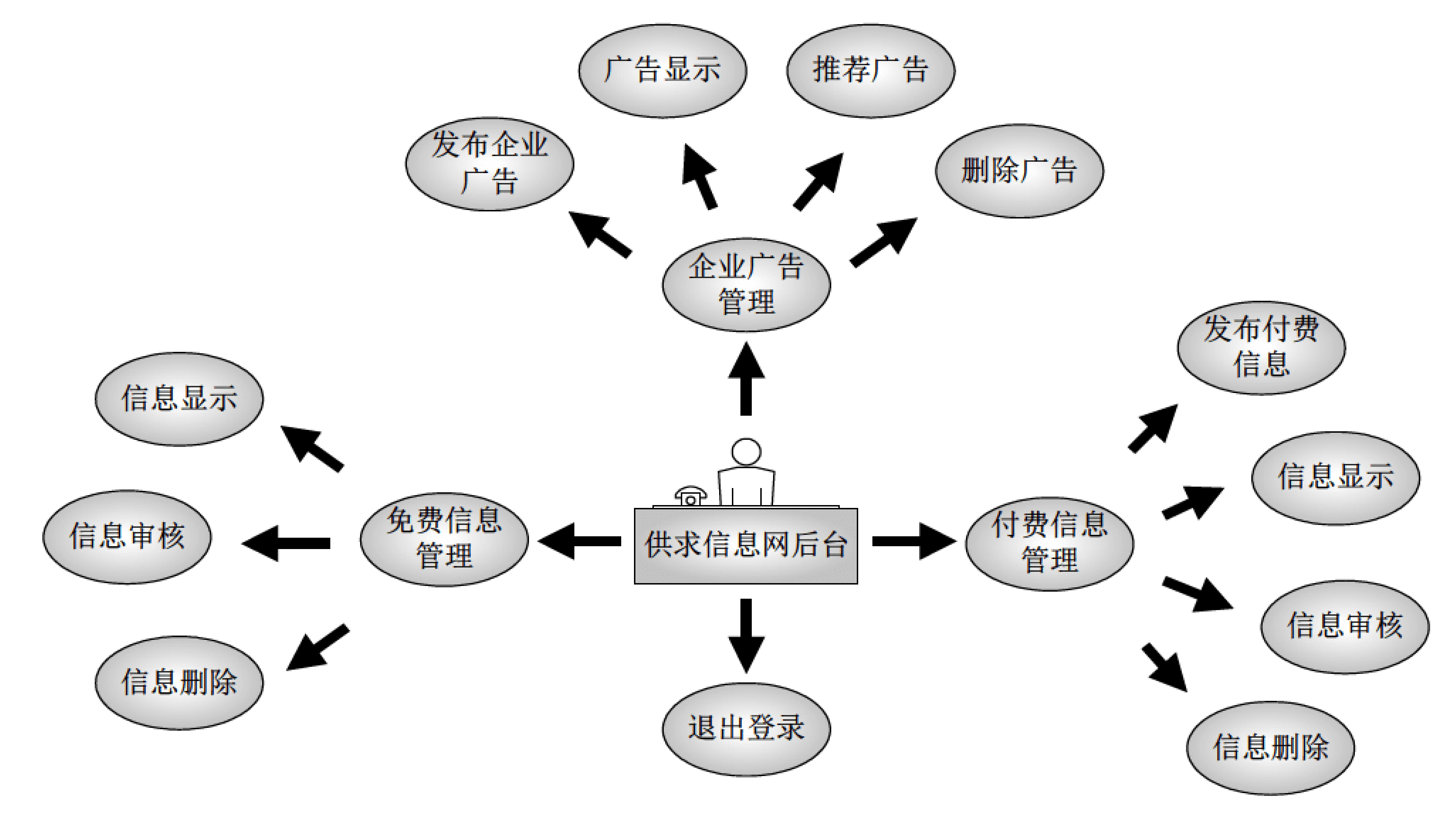


图 27.2　易查供求信息网后台功能结构图

易查供求信息网的系统流程如图 27.3 所示。

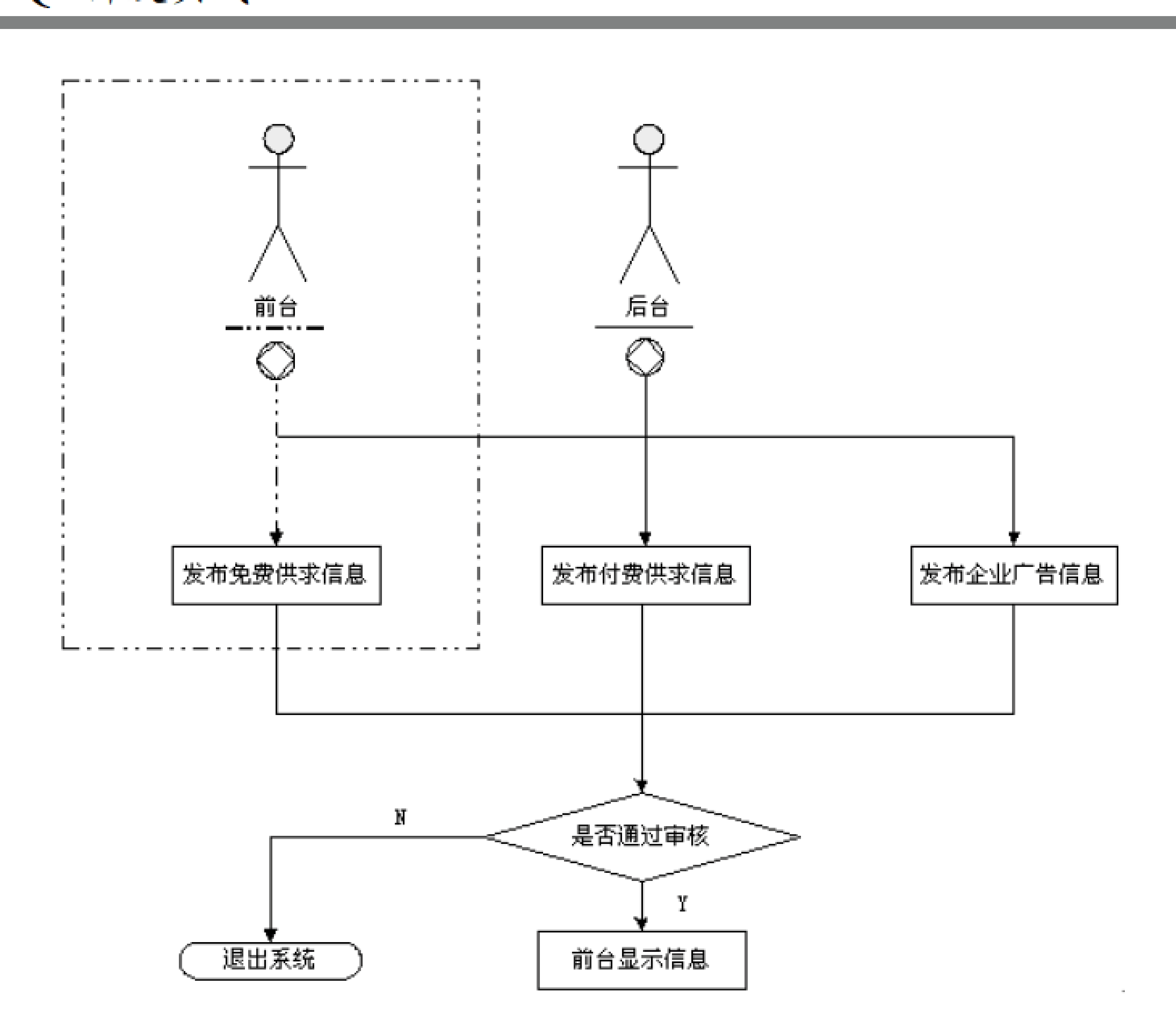


图 27.3　系统流程图

27.3.3　系统预览

易查供求信息网由多个程序页面组成。下面列出几个典型页面，其他页面参见光盘中的源程序。

前台首页如图 27.4 所示，该页面用于实现各类信息的查询、企业广告信息显示、后台登录入口等功能。搜索引擎页面如图 27.5 所示，该页面用于实现各类信息的快速检索、查询关键字描红等功能。发布免费信息页面如图 27.6 所示，该页面用于实现发布分类的免费信息功能。付费管理页面如图 27.7 所示，该页面用于实现付费信息分类查看、付费信息审核、付费信息删除等功能。



图 27.4　前台首页



图 27.5　搜索引擎



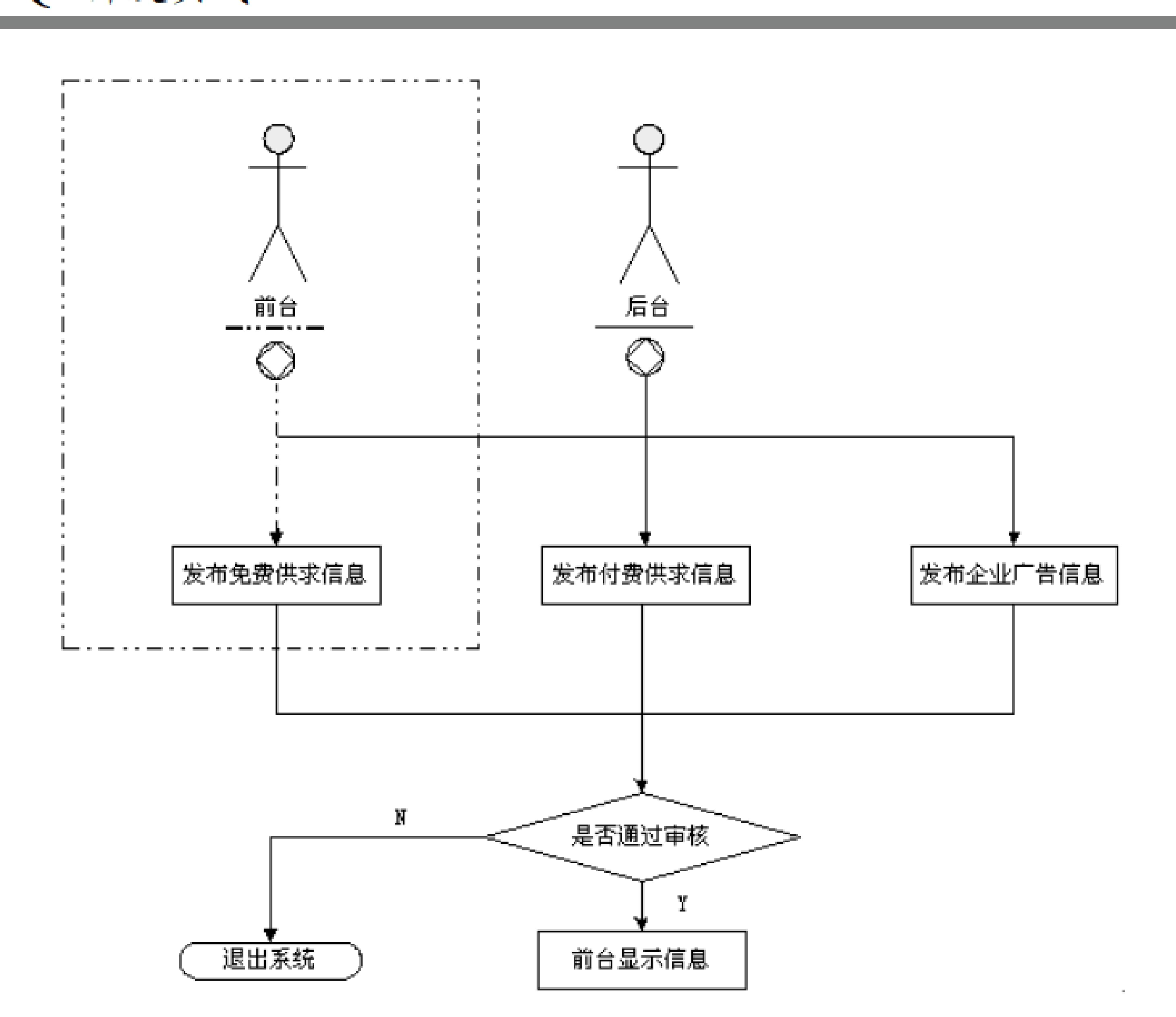


图 27.3　系统流程图

27.3.3　系统预览

易查供求信息网由多个程序页面组成。下面列出几个典型页面，其他页面参见光盘中的源程序。

前台首页如图 27.4 所示，该页面用于实现各类信息的查询、企业广告信息显示、后台登录入口等功能。搜索引擎页面如图 27.5 所示，该页面用于实现各类信息的快速检索、查询关键字描红等功能。发布免费信息页面如图 27.6 所示，该页面用于实现发布分类的免费信息功能。付费管理页面如图 27.7 所示，该页面用于实现付费信息分类查看、付费信息审核、付费信息删除等功能。



图 27.4　前台首页



图 27.5　搜索引擎





图 27.6　发布免费信息



图 27.7　付费管理

免费管理页面如图 27.8 所示，该页面用于实现免费信息分类查看、免费信息审核、免费信息删除等功能。管理员登录页面如图 27.9 所示，该页面用于实现对管理员登录的用户名和密码进行验证等功能。



图 27.8　免费管理



图 27.9　管理员登录

27.3.4　文件夹组织结构

视频讲解：光盘\TM\Video\第 27 章\文件夹组织结构.exe

在编写代码前，可以把系统中可能用到的文件夹先创建出来（如创建一个名为 admin 的文件夹，用于保存网站的后台文件），这样不但可以方便以后的开发工作，也可以规范网站的整体架构。笔者在开发易查供求信息网时，设计了如图 27.10 所示的文件夹架构图。在开发时，只需要将所创建的文件保存在相应的文件夹中即可。

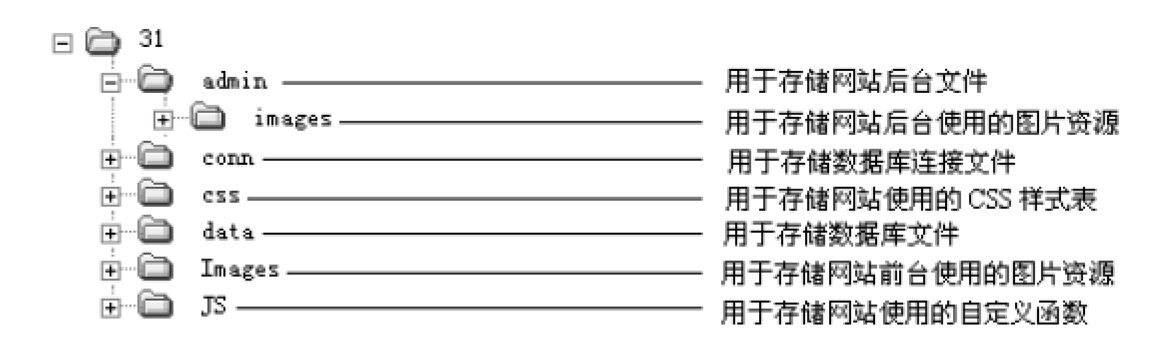


图 27.10　文件夹组织结构

27.4　数据库设计

视频讲解：光盘\TM\Video\第 27 章\数据库设计.exe

27.4.1　数据库分析

本系统是一个中小型的供求信息平台，但是由于平台会涉及到海量数据，因此需要充分考虑到成本问题及使用需求（如跨平台）等问题，而 MySQL 是世界上最为流行的开放源代码的数据库，是完全网络化的、跨平台的关系型数据库系统，这正好满足了中小型企业的需求，所以本系统采用 MySQL 数据库。

27.4.2　数据库概念设计

根据前面对系统所做的需求分析、系统设计，规划出本系统中使用的数据库实体分别为免费信息实体、付费信息实体、广告信息实体、管理员实体。下面将介绍这几个实体的 E-R 图。

1．免费信息实体

免费信息实体包括编号、信息类型、信息标题、信息内容、联系人、联系电话、审核状态和发布时间属性。其中审核状态属性用来标识信息是否审核，“1”表示“是”，“0”表示“否”。免费信息实体的 E-R 图如图 27.11 所示。

2．付费信息实体

付费信息实体包括编号、信息类型、信息标题、信息内容、联系人、联系电话、发布时间、截止时间和审核状态属性。其中审核状态属性用来标识信息是否付费，“1”表示“是”，“0”表示“否”。付费信息实体的 E-R 图如图 27.12 所示。

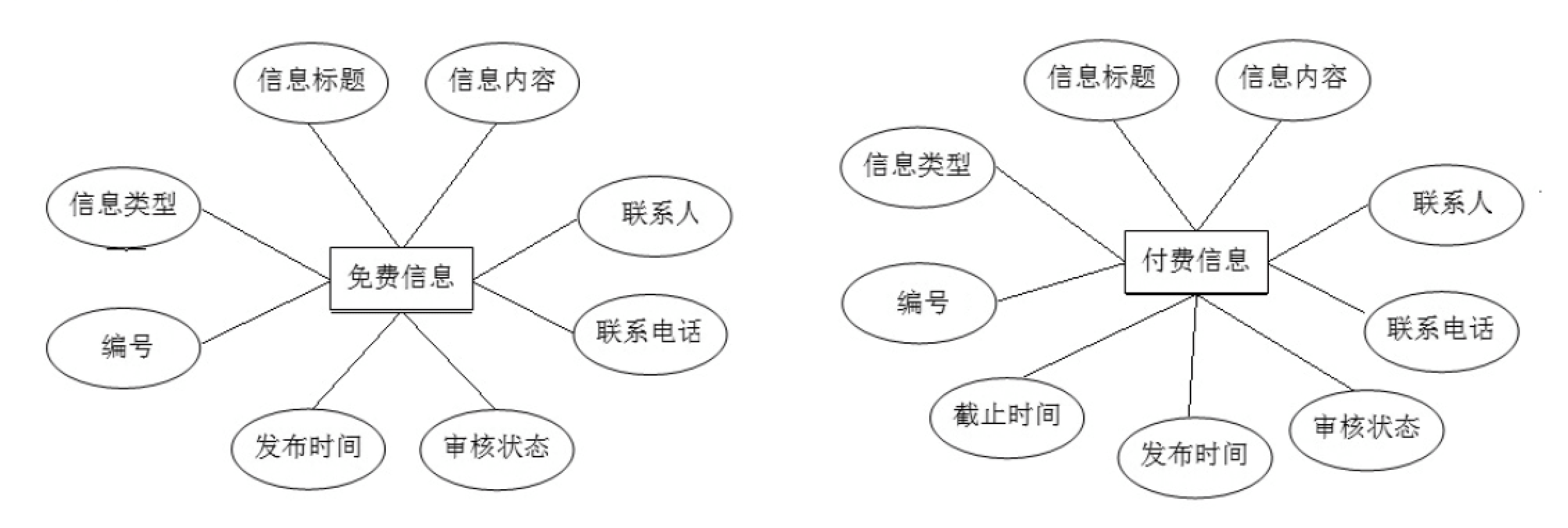


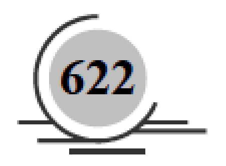
图 27.11　免费信息实体 E-R 图　　　图 27.12　付费信息实体的 E-R 图

3．广告信息实体

广告信息实体包括编号、信息标题、信息内容、发布时间和推荐状态属性。其中推荐状态属性用来标识信息是否在前台显示，“1”表示“是”，“0”表示“否”。广告信息实体的 E-R 图如图 27.13 所示。

4．管理员实体

管理员实体包括编号、管理员名和加密密码属性。管理员实体的 E-R 图如图 27.14 所示。



27.4 数据库设计

视频讲解：光盘\TM\Video\第 27 章\数据库设计.exe

27.4.1 数据库分析

本系统是一个中小型的供求信息平台，但是由于平台会涉及到海量数据，因此需要充分考虑到成本问题及使用需求（如跨平台）等问题，而 MySQL 是世界上最为流行的开放源代码的数据库，是完全网络化的、跨平台的关系型数据库系统，这正好满足了中小型企业的需求，所以本系统采用 MySQL 数据库。

27.4.2 数据库概念设计

根据前面对系统所做的需求分析、系统设计，规划出本系统中使用的数据库实体分别为免费信息实体、付费信息实体、广告信息实体、管理员实体。下面将介绍这几个实体的 E-R 图。

1. 免费信息实体

免费信息实体包括编号、信息类型、信息标题、信息内容、联系人、联系电话、审核状态和发布时间属性。其中审核状态属性用来标识信息是否审核，“1”表示“是”，“0”表示“否”。免费信息实体的 E-R 图如图 27.11 所示。

2. 付费信息实体

付费信息实体包括编号、信息类型、信息标题、信息内容、联系人、联系电话、发布时间、截止时间和审核状态属性。其中审核状态属性用来标识信息是否付费，“1”表示“是”，“0”表示“否”。付费信息实体的 E-R 图如图 27.12 所示。

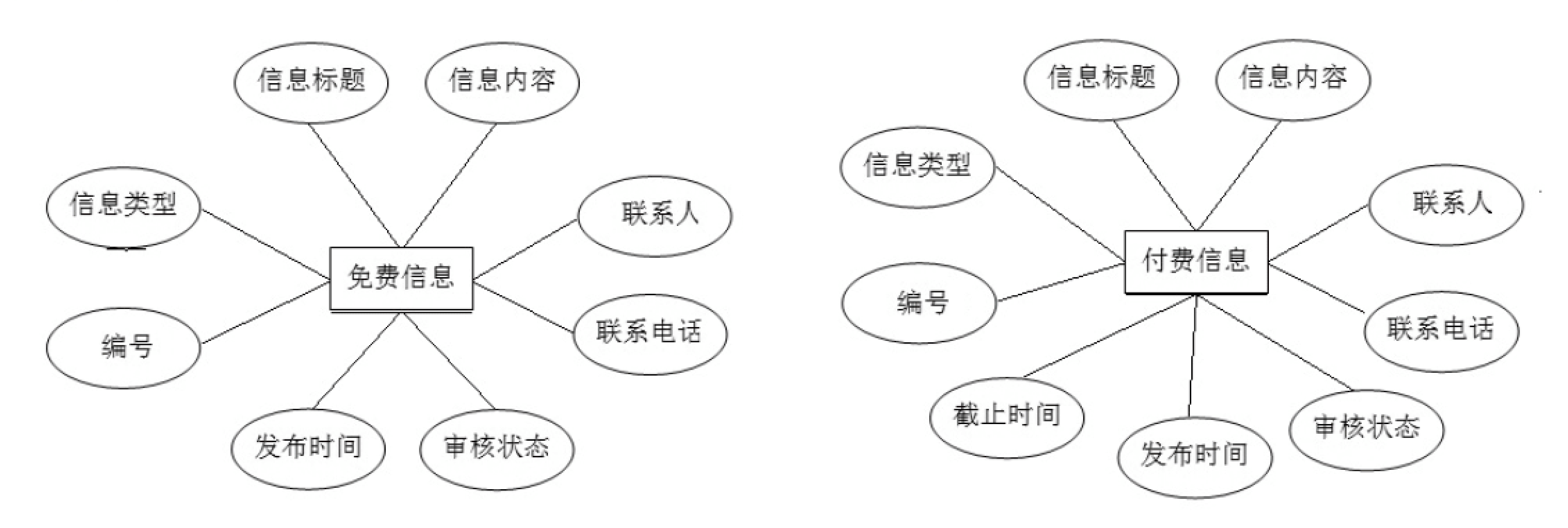


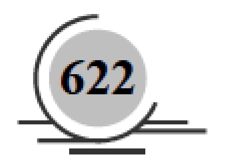
图 27.11 免费信息实体 E-R 图　　图 27.12 付费信息实体的 E-R 图

3. 广告信息实体

广告信息实体包括编号、信息标题、信息内容、发布时间和推荐状态属性。其中推荐状态属性用来标识信息是否在前台显示，“1”表示“是”，“0”表示“否”。广告信息实体的 E-R 图如图 27.13 所示。

4. 管理员实体

管理员实体包括编号、管理员名和加密密码属性。管理员实体的 E-R 图如图 27.14 所示。



27.4 数据库设计

视频讲解：光盘\TM\Video\第 27 章\数据库设计.exe

27.4.1 数据库分析

本系统是一个中小型的供求信息平台，但是由于平台会涉及到海量数据，因此需要充分考虑到成本问题及使用需求（如跨平台）等问题，而 MySQL 是世界上最为流行的开放源代码的数据库，是完全网络化的、跨平台的关系型数据库系统，这正好满足了中小型企业的需求，所以本系统采用 MySQL 数据库。

27.4.2 数据库概念设计

根据前面对系统所做的需求分析、系统设计，规划出本系统中使用的数据库实体分别为免费信息实体、付费信息实体、广告信息实体、管理员实体。下面将介绍这几个实体的 E-R 图。

1．免费信息实体

免费信息实体包括编号、信息类型、信息标题、信息内容、联系人、联系电话、审核状态和发布时间属性。其中审核状态属性用来标识信息是否审核，“1”表示“是”，“0”表示“否”。免费信息实体的 E-R 图如图 27.11 所示。

2．付费信息实体

付费信息实体包括编号、信息类型、信息标题、信息内容、联系人、联系电话、发布时间、截止时间和审核状态属性。其中审核状态属性用来标识信息是否付费，“1”表示“是”，“0”表示“否”。付费信息实体的 E-R 图如图 27.12 所示。

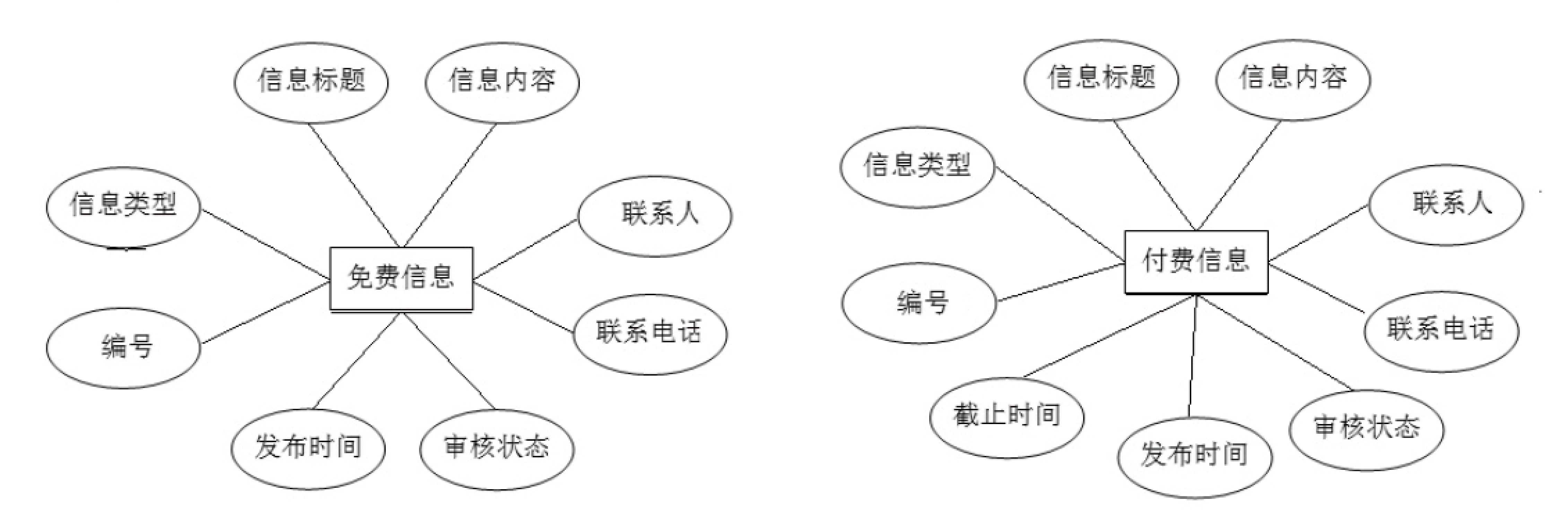


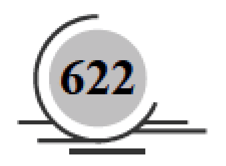
图 27.11 免费信息实体 E-R 图　　　　图 27.12 付费信息实体的 E-R 图

3．广告信息实体

广告信息实体包括编号、信息标题、信息内容、发布时间和推荐状态属性。其中推荐状态属性用来标识信息是否在前台显示，“1”表示“是”，“0”表示“否”。广告信息实体的 E-R 图如图 27.13 所示。

4．管理员实体

管理员实体包括编号、管理员名和加密密码属性。管理员实体的 E-R 图如图 27.14 所示。



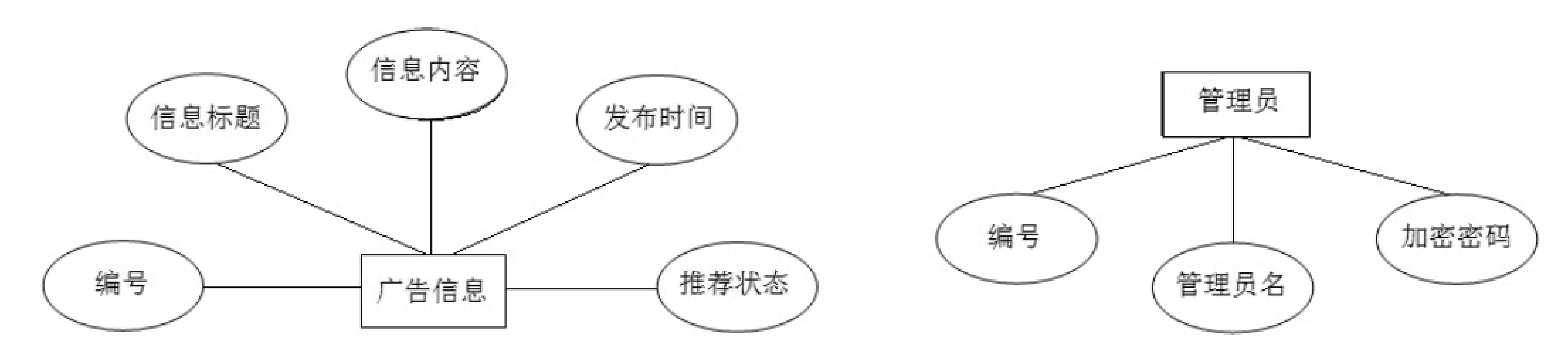


图 27.13　广告信息实体 E-R 图　　图 27.14　管理员实体 E-R 图

27.4.3　创建数据库及数据表

结合实际情况及对用户需求的分析，可知易查供求信息网中应用的 db_pursey 数据库主要包含如下 4 个数据表，如图 27.15 所示。

服务器: localhost ▸ 数据库: db_pursey

表	类型	整理	说明
tb_admin	MyISAM	gb2312_chinese_ci	管理员信息表
tb_advertising	MyISAM	gb2312_chinese_ci	企业广告信息表
tb_info	MyISAM	gb2312_chinese_ci	免费供求信息表
tb_leaguerinfo	MyISAM	gb2312_chinese_ci	付费供求信息表

图 27.15　易查供求信息网数据表

各数据表的表结构如图 27.16～图 27.19 所示。

1．tb_admin（管理员信息表）

管理员信息表主要用于存储管理员的信息。该数据表的结构如图 27.16 所示。

2．tb_advertising（企业广告信息表）

企业广告信息表主要用于存储企业发布的广告信息。该数据表的结构如图 27.17 所示。

服务器: localhost ▸ 数据库: db_pursey ▸ 表 : tb_admin

字段	类型	整理	Null	默认	额外	说明
id	int(4)		否		auto_increment	自动编号
name	varchar(50)	gb2312_chinese_ci	是	NULL		管理员名
pwd	varchar(50)	gb2312_chinese_ci	是	NULL		管理员密码

图 27.16　管理员信息表结构



服务器: localhost ▸ 数据库: db_pursey ▸ 表 : tb_advertising

字段	类型	整理	Null	默认	额外	说明
id	int(4)		否		auto_increment	自动编号
title	varchar(100)	gb2312_chinese_ci	否			广告标题
content	varchar(500)	gb2312_chinese_ci	否			广告内容
fdate	datetime		否			发布时间
flag	int(1)		否	0		推荐状态

图 27.17　企业广告信息表结构

3．tb_info（免费供求信息表）

免费供求信息表主要用于存储用户免费发布的供求信息。该数据表的结构如图 27.18 所示。

4．tb_leaguerinfo（付费供求信息表）

付费供求信息表主要用于存储付费的供求信息。该数据表的结构如图 27.19 所示。

服务器: localhost ▸ 数据库: db_pursey ▸ 表 : tb_info

字段	类型	整理	Null	默认	额外	说明
id	int(4)		否		auto_increment	自动编号
type	varchar(30)	gb2312_chinese_ci	否			信息类型
title	varchar(50)	gb2312_chinese_ci	否			信息标题
content	varchar(500)	gb2312_chinese_ci	否			信息内容
linkman	varchar(20)	gb2312_chinese_ci	否			联系人
tel	varchar(30)	gb2312_chinese_ci	否			联系电话
checkstate	int(1)		否	0		审核状态
edate	datetime		否			发布时间

图 27.18　免费供求信息表结构



服务器: localhost ▸ 数据库: db_pursey ▸ 表 :tb_leaguerinfo

字段	类型	整理	Null	默认	额外	说明
id	int(4)		否		auto_increment	自动编号
type	varchar(20)	gb2312_chinese_ci	否			信息类型
title	varchar(50)	gb2312_chinese_ci	否			信息标题
content	varchar(500)	gb2312_chinese_ci	否			信息内容
linkman	varchar(20)	gb2312_chinese_ci	否			联系人
tel	varchar(30)	gb2312_chinese_ci	否			联系电话
sdate	date		否			发布日期
showday	date		否			截止日期
checkstate	int(1)		否	0		审核状态

图 27.19　付费供求信息表结构

27.5 前台首页设计

视频讲解：光盘\TM\Video\第 27 章\前台首页设计.exe

27.5.1 前台首页概述

网站主页是关于网站的建设及形象宣传，它对网站的生存和发展起着非常重要的作用，应该是一个信息含量较高、内容较丰富的宣传平台。易查供求信息网前台首页主要包含以下内容。

- ☑ 网站菜单导航（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等）。
- ☑ 发布免费的供求信息（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等）。
- ☑ 推荐供求信息显示（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等），其中，付费信息按时间顺序降序排列，免费信息按时间顺序分页显示。
- ☑ 显示推荐的企业广告信息。
- ☑ 供求信息快速检索：支持模糊查询和查询关键字描红功能。
- ☑ 后台登录入口：为管理员进入后台提供一个入口。

下面看一下本案例中提供的前台首页，该页面在本书光盘中的路径为 TM\27\index.php，如图 27.20 所示。



图 27.20 易查供求信息网首页

27.5　前台首页设计

视频讲解：光盘\TM\Video\第 27 章\前台首页设计.exe

27.5.1　前台首页概述

网站主页是关于网站的建设及形象宣传，它对网站的生存和发展起着非常重要的作用，应该是一个信息含量较高、内容较丰富的宣传平台。易查供求信息网前台首页主要包含以下内容。

- ☑ 网站菜单导航（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等）。
- ☑ 发布免费的供求信息（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等）。
- ☑ 推荐供求信息显示（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等），其中，付费信息按时间顺序降序排列，免费信息按时间顺序分页显示。
- ☑ 显示推荐的企业广告信息。
- ☑ 供求信息快速检索：支持模糊查询和查询关键字描红功能。
- ☑ 后台登录入口：为管理员进入后台提供一个入口。

下面看一下本案例中提供的前台首页，该页面在本书光盘中的路径为 TM\27\index.php，如图 27.20 所示。



图 27.20　易查供求信息网首页

27.5 前台首页设计

视频讲解：光盘\TM\Video\第 27 章\前台首页设计.exe

27.5.1 前台首页概述

网站主页是关于网站的建设及形象宣传，它对网站的生存和发展起着非常重要的作用，应该是一个信息含量较高、内容较丰富的宣传平台。易查供求信息网前台首页主要包含以下内容。

- ☑ 网站菜单导航（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等）。
- ☑ 发布免费的供求信息（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等）。
- ☑ 推荐供求信息显示（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等），其中，付费信息按时间顺序降序排列，免费信息按时间顺序分页显示。
- ☑ 显示推荐的企业广告信息。
- ☑ 供求信息快速检索：支持模糊查询和查询关键字描红功能。
- ☑ 后台登录入口：为管理员进入后台提供一个入口。

下面看一下本案例中提供的前台首页，该页面在本书光盘中的路径为 TM\27\index.php，如图 27.20 所示。



图 27.20 易查供求信息网首页

各区域的介绍及所对应的 PHP 文件如表 27.1 所示。

表 27.1　页面框架中各区域介绍及对应 PHP 文件

名　称	说　明	对应 PHP 文件
导航栏	提供查看各类信息的超链接	top.php
信息检索区	企业广告显示及各类信息检索	left.php
内容显示区	根据用户请求显示相应内容	根据请求加载相应的 PHP 文件，默认加载 main.php
版权区	显示版权信息	bottom.php

27.5.2　超链接技术

超链接在本质上属于一个网页的一部分，它是一种允许用户同其他网页或站点之间进行连接的元素。各个网页链接在一起后，才能真正构成一个网站。按照使用对象的不同，网页中的链接可以分为文本超链接、图像超链接、E-mail 链接、锚点链接、多媒体文件链接和空链接等。

图像不但可以建立超链接，还可以实现图像映射。图像映射是指一幅图像可以建立多个超链接，即在图像上定义多个区域，每个区域链接到不同的地址，这样的区域称为热区（也称为热点，Hotspot）。本项目就是应用图像热区实现的功能导航。

图像映射有两种：服务器端映射（Server-side Image Map）和客户端映射（Client-side Image Map）。目前使用最多的是客户端映射，因为客户端映射使图像上对应区域的坐标以及链接的 URL 地址都在浏览器端读入，省去和服务器之间互传坐标和 URL 的时间。

在 PHP 文件中，使用<MAP>标记创建图像映射。语法如下：

```
<IMG SRC="file_name" USEMAP="#MapName">
<MAP NAME="MapName">
<AREA SHAPE="value" COORDS="坐标"HREF="URL" ALT="描述文字">
<AREA SHAPE="value" COORDS="坐标"HREF="URL" ALT="描述文字">
…
</MAP>
```

在<IMG>标记中设置属性 USEMAP，确定创建图像映射。<MAP>标记的属性如表 27.2 所示。

表 27.2　<MAP>标记的属性

<MAP>标记的属性	描　述
NAME	图像映射的名称
SHAPE	定义图像映射区域的名称
COORDS	设定区域坐标
HREF	设定区域的链接地址
ALT	设定区域链接的描述文字

在<MAP>标记中，根据属性 SHAPE 的取值不同，相应坐标的设定也不同。下面介绍属性 SHAPE 的 3 种取值以及相应坐标的设定。

- ☑ 设定属性 SHAPE 的属性值为 rect。属性 SHAPE 取值为 rect，表示矩形区域，属性 COORDS 的坐标形式为“X1,Y1,X2,Y2”。其中，X1、Y1 代表矩形左上角的 X、Y 坐标，X2、Y2 代表矩形右下角的 X、Y 坐标。
- ☑ 设定属性 SHAPE 的属性值为 circle。属性 SHAPE 取值为 circle，表示圆形区域，属性 COORDS 的坐标形式为“X,Y,r”。其中，X、Y 为圆心坐标，r 为圆的半径。

各区域的介绍及所对应的 PHP 文件如表 27.1 所示。

表 27.1　页面框架中各区域介绍及对应 PHP 文件

名　　称	说　　明	对应 PHP 文件
导航栏	提供查看各类信息的超链接	top.php
信息检索区	企业广告显示及各类信息检索	left.php
内容显示区	根据用户请求显示相应内容	根据请求加载相应的 PHP 文件，默认加载 main.php
版权区	显示版权信息	bottom.php

27.5.2　超链接技术

超链接在本质上属于一个网页的一部分，它是一种允许用户同其他网页或站点之间进行连接的元素。各个网页链接在一起后，才能真正构成一个网站。按照使用对象的不同，网页中的链接可以分为文本超链接、图像超链接、E-mail 链接、锚点链接、多媒体文件链接和空链接等。

图像不但可以建立超链接，还可以实现图像映射。图像映射是指一幅图像可以建立多个超链接，即在图像上定义多个区域，每个区域链接到不同的地址，这样的区域称为热区（也称为热点，Hotspot）。本项目就是应用图像热区实现的功能导航。

图像映射有两种：服务器端映射（Server-side Image Map）和客户端映射（Client-side Image Map）。目前使用最多的是客户端映射，因为客户端映射使图像上对应区域的坐标以及链接的 URL 地址都在浏览器端读入，省去和服务器之间互传坐标和 URL 的时间。

在 PHP 文件中，使用<MAP>标记创建图像映射。语法如下：

```
<IMG SRC="file_name" USEMAP="#MapName">
<MAP NAME="MapName">
<AREA SHAPE="value" COORDS="坐标"HREF="URL" ALT="描述文字">
<AREA SHAPE="value" COORDS="坐标"HREF="URL" ALT="描述文字">
…
</MAP>
```

在<IMG>标记中设置属性 USEMAP，确定创建图像映射。<MAP>标记的属性如表 27.2 所示。

表 27.2　<MAP>标记的属性

<MAP>标记的属性	描　　述
NAME	图像映射的名称
SHAPE	定义图像映射区域的名称
COORDS	设定区域坐标
HREF	设定区域的链接地址
ALT	设定区域链接的描述文字

在<MAP>标记中，根据属性 SHAPE 的取值不同，相应坐标的设定也不同。下面介绍属性 SHAPE 的 3 种取值以及相应坐标的设定。

☑ 设定属性 SHAPE 的属性值为 rect。属性 SHAPE 取值为 rect，表示矩形区域，属性 COORDS 的坐标形式为“X1,Y1,X2,Y2”。其中，X1、Y1 代表矩形左上角的 X、Y 坐标，X2、Y2 代表矩形右下角的 X、Y 坐标。

☑ 设定属性 SHAPE 的属性值为 circle。属性 SHAPE 取值为 circle，表示圆形区域，属性 COORDS 的坐标形式为“X,Y,r”。其中，X、Y 为圆心坐标，r 为圆的半径。

各区域的介绍及所对应的 PHP 文件如表 27.1 所示。

表 27.1　页面框架中各区域介绍及对应 PHP 文件

名　称	说　明	对应 PHP 文件
导航栏	提供查看各类信息的超链接	top.php
信息检索区	企业广告显示及各类信息检索	left.php
内容显示区	根据用户请求显示相应内容	根据请求加载相应的 PHP 文件，默认加载 main.php
版权区	显示版权信息	bottom.php

27.5.2　超链接技术

超链接在本质上属于一个网页的一部分，它是一种允许用户同其他网页或站点之间进行连接的元素。各个网页链接在一起后，才能真正构成一个网站。按照使用对象的不同，网页中的链接可以分为文本超链接、图像超链接、E-mail 链接、锚点链接、多媒体文件链接和空链接等。

图像不但可以建立超链接，还可以实现图像映射。图像映射是指一幅图像可以建立多个超链接，即在图像上定义多个区域，每个区域链接到不同的地址，这样的区域称为热区（也称为热点，Hotspot）。本项目就是应用图像热区实现的功能导航。

图像映射有两种：服务器端映射（Server-side Image Map）和客户端映射（Client-side Image Map）。目前使用最多的是客户端映射，因为客户端映射使图像上对应区域的坐标以及链接的 URL 地址都在浏览器端读入，省去和服务器之间互传坐标和 URL 的时间。

在 PHP 文件中，使用<MAP>标记创建图像映射。语法如下：

```
<IMG SRC="file_name" USEMAP="#MapName">
<MAP NAME="MapName">
<AREA SHAPE="value" COORDS="坐标"HREF="URL" ALT="描述文字">
<AREA SHAPE="value" COORDS="坐标"HREF="URL" ALT="描述文字">
…
</MAP>
```

在<IMG>标记中设置属性 USEMAP，确定创建图像映射。<MAP>标记的属性如表 27.2 所示。

表 27.2　<MAP>标记的属性

<MAP>标记的属性	描　述
NAME	图像映射的名称
SHAPE	定义图像映射区域的名称
COORDS	设定区域坐标
HREF	设定区域的链接地址
ALT	设定区域链接的描述文字

在<MAP>标记中，根据属性 SHAPE 的取值不同，相应坐标的设定也不同。下面介绍属性 SHAPE 的 3 种取值以及相应坐标的设定。

- ☑ 设定属性 SHAPE 的属性值为 rect。属性 SHAPE 取值为 rect，表示矩形区域，属性 COORDS 的坐标形式为“X1,Y1,X2,Y2”。其中，X1、Y1 代表矩形左上角的 X、Y 坐标，X2、Y2 代表矩形右下角的 X、Y 坐标。
- ☑ 设定属性 SHAPE 的属性值为 circle。属性 SHAPE 取值为 circle，表示圆形区域，属性 COORDS 的坐标形式为“X,Y,r”。其中，X、Y 为圆心坐标，r 为圆的半径。

☑ 设定属性 SHAPE 的属性值为 poly。属性 SHAPE 取值为 poly，表示多边形区域，属性 COORDS 的坐标形式为“X1,Y1,X2,Y2,…”。其中，Xn、Yn 代表构成多边形每一点的坐标值，n 的取值为 1，2，3，…。多边形有几边就有几对 X、Y 坐标。

下面在 Dreamweaver 编辑器中创建图像映射，单击图像不同区域，链接到不同的文件。创建图像映射的步骤如下：

（1）在 Dreamweaver 编辑器中选中所要编辑的图像，在属性窗口中，有 3 种形状的热点工具，分别为矩形热点工具、圆形热点工具和多边形热点工具。

（2）选择一种热点工具，在图像上按住鼠标左键拖动，确定所选区域。

（3）设置区域属性，在“链接”文本框中输入所要链接书签的名称，在“替代”文本框中输入区域链接文件，如 index.php。

（4）重复步骤（2）、（3），在图像上定义不同的区域链接。

27.5.3　前台首页的实现过程

本系统中所有的前台页面都采用了二分栏结构，分为导航栏、信息检索区、内容显示区和版权区 4 个区域。为了方便网站的日后维护，将这 4 个区域形成单独的 PHP 文件，然后应用 include 语句将这 4 个文件包含进来。前台首页文件的代码如下：

（代码位置：光盘\TM\Instance\27\index.php）

```
<table width="780" border="0" align="center" cellpadding="0" cellspacing="0">
  <tr valign="top">
    <td colspan="2"><?php include("top.php");?></td>                    <!-- 包含导航文件 -->
  </tr>
  <tr>
     <!-- 包含信息检索文件 -->
    <td width="217" valign="top" background="Images/line2.gif"><?php include("left.php");?></td>
     <!-- 包含内容显示文件 -->
    <td width="586" valign="top" bgcolor="#FEFEF6"><?php include("main.php");?></td>
  </tr>
  <tr>
    <td colspan="2"><?php include("bottom.php");?></td>                 <!-- 包含版权信息文件 -->
  </tr>
</table>
```

其中，导航文件 top.php 页应用了图像映射来创建文件的超链接。其代码如下：

（代码位置：光盘\TM\Instance\27\top.php）

```
<table width="780" height="201" border="0" align="center" cellpadding="0" cellspacing="0">
  <tr>
    <td height="38"><img src="Images/banner.gif" width="780" height="202" border="0" usemap="#Map"></td>
  </tr>
</table>
<!-- -----------------------------------------------图像映射------------------------------------ -->
<map name="Map">
<!-- -------------------------------------------联系我们------------------------------------ -->
  <area shape="rect" coords="685,8,744,27" href="mailto:99pursey@pursey**.com">
<!-- -------------------------------------------设为首页------------------------------------ -->
  <area shape="rect" coords="6163,9,666,27"
onClick="this.style.behavior='url(#default#homepage)';this.setHomePage('http://localhost/99pursey/index.php');
">
<!-- -------------------------------------------加入收藏------------------------------------- -->
  <area shape="rect" coords="535,8,593,26"
href="javascript:window.external.AddFavorite('http://localhost/99pursey/index.php','易查供求信息网')">
```

```
    <area shape="rect" coords="448,8,516,26" href="admin/login.php">              <!--  后台登录页  -->
    <area shape="rect" coords="356,8,424,28" href="release.php">                  <!--  发布信息页  -->
    <area shape="rect" coords="680,61,753,82" href="search.php">                  <!--  寻人/物启示信息页  -->
    <area shape="rect" coords="579,62,637,82" href="sale.php">                    <!--  出售信息页  -->
    <area shape="rect" coords="483,62,536,83" href="car.php">                     <!--  车辆信息页  -->
    <area shape="rect" coords="385,62,441,83" href="teaching.php">                <!--  家教信息页  -->
    <area shape="rect" coords="290,61,344,83" href="seekjob.php">                 <!--  求职信息页  -->
    <area shape="rect" coords="678,38,740,58" href="recruitbusiness.php">         <!--  招商引资信息页  -->
    <area shape="rect" coords="579,38,635,58" href="seekbuy.php">                 <!--  求购信息页  -->
    <area shape="rect" coords="482,38,541,58" href="house.php">                   <!--  房屋信息页  -->
    <area shape="rect" coords="381,39,440,60" href="foster.php">                  <!--  培训信息页  -->
    <area shape="rect" coords="290,39,345,59" href="invitejob.php">               <!--  招聘信息页  -->
    <area shape="rect" coords="204,39,263,86" href="index.php">                   <!--  首页  -->
</map>
```

27.6　免费供求信息发布模块设计

视频讲解：光盘\TM\Video\第 26 章\免费供求信息发布模块设计.exe

27.6.1　免费供求信息发布模块概述

免费供求信息发布功能是供求信息网站最基本、最核心的功能。

免费供求信息发布模块可以完成 11 种不同类别信息的发布。用户可以根据自身需要将供求信息发布到相应的信息类别中（共包括 11 个信息类别：公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示）。信息发布成功后，需要管理员进行审核，只有审核成功的信息才能显示在前台相应的信息类别网页中。免费供求信息发布的流程如图 27.21 所示。

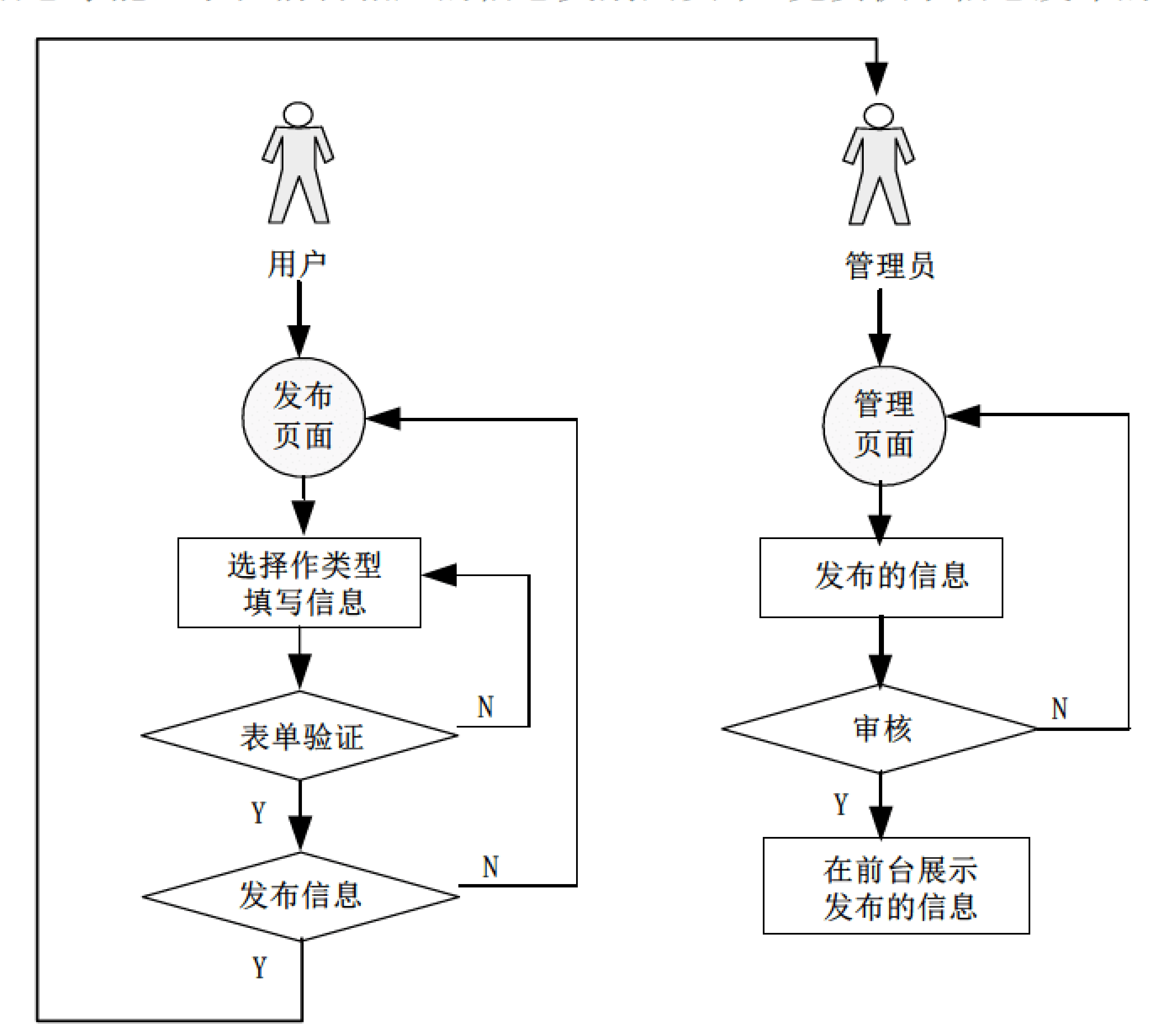


图 27.21　免费供求信息发布流程图

```
  <area shape="rect" coords="448,8,516,26" href="admin/login.php">          <!-- 后台登录页 -->
  <area shape="rect" coords="356,8,424,28" href="release.php">              <!-- 发布信息页 -->
  <area shape="rect" coords="680,61,753,82" href="search.php">              <!-- 寻人/物启示信息页 -->
  <area shape="rect" coords="579,62,637,82" href="sale.php">                <!-- 出售信息页 -->
  <area shape="rect" coords="483,62,536,83" href="car.php">                 <!-- 车辆信息页 -->
  <area shape="rect" coords="385,62,441,83" href="teaching.php">            <!-- 家教信息页 -->
  <area shape="rect" coords="290,61,344,83" href="seekjob.php">             <!-- 求职信息页 -->
  <area shape="rect" coords="678,38,740,58" href="recruitbusiness.php">     <!-- 招商引资信息页 -->
  <area shape="rect" coords="579,38,635,58" href="seekbuy.php">             <!-- 求购信息页 -->
  <area shape="rect" coords="482,38,541,58" href="house.php">               <!-- 房屋信息页 -->
  <area shape="rect" coords="381,39,440,60" href="foster.php">              <!-- 培训信息页 -->
  <area shape="rect" coords="290,39,345,59" href="invitejob.php">           <!-- 招聘信息页 -->
  <area shape="rect" coords="204,39,263,86" href="index.php">               <!-- 首页 -->
</map>
```

27.6　免费供求信息发布模块设计

视频讲解：光盘\TM\Video\第 26 章\免费供求信息发布模块设计.exe

27.6.1　免费供求信息发布模块概述

免费供求信息发布功能是供求信息网站最基本、最核心的功能。

免费供求信息发布模块可以完成 11 种不同类别信息的发布。用户可以根据自身需要将供求信息发布到相应的信息类别中（共包括 11 个信息类别：公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示）。信息发布成功后，需要管理员进行审核，只有审核成功的信息才能显示在前台相应的信息类别网页中。免费供求信息发布的流程如图 27.21 所示。

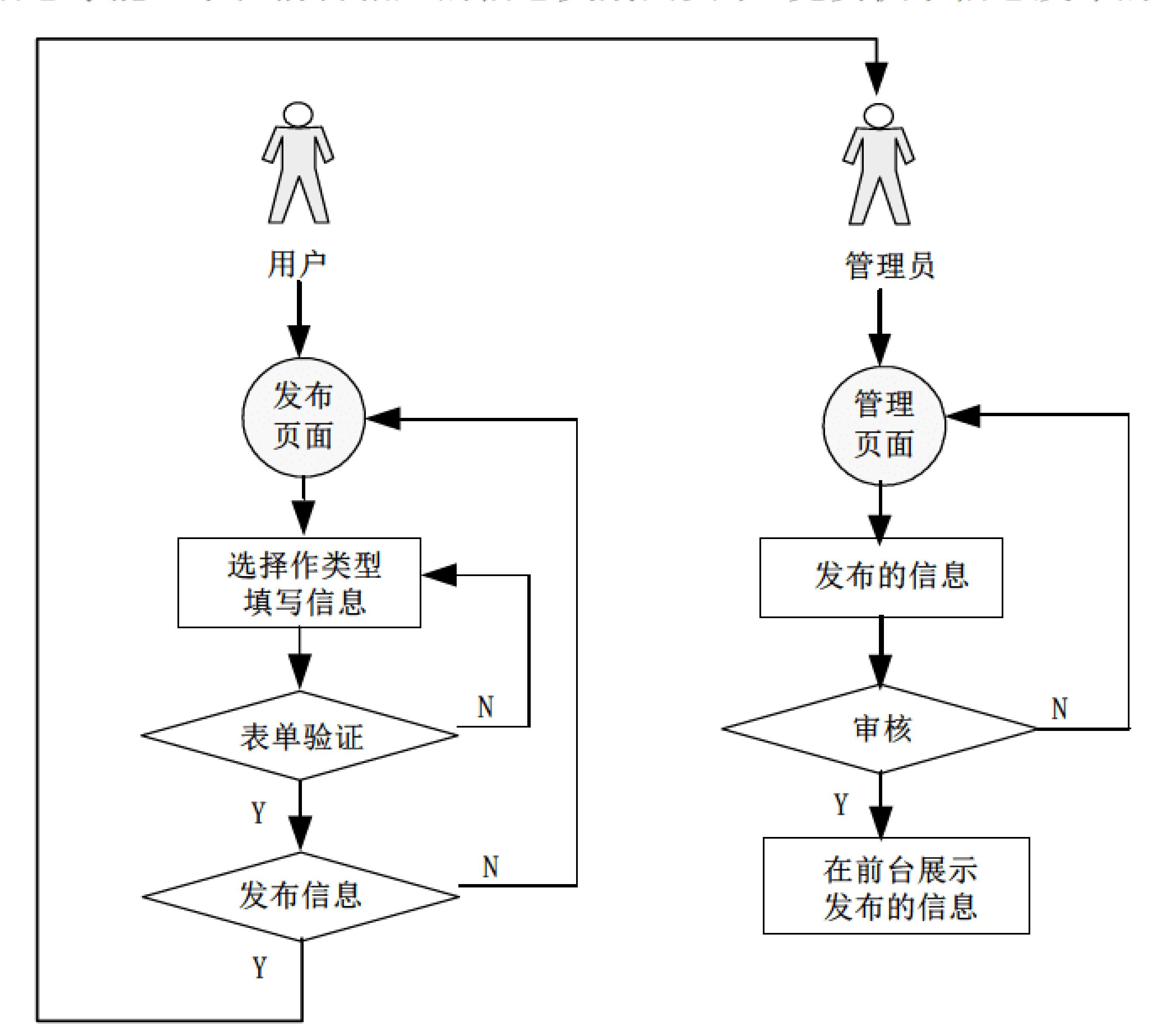


图 27.21　免费供求信息发布流程图

27.6.2 MySQL 数据库连接技术

本模块实现免费供求信息发布功能主要应用到如下几个函数，下面进行详细介绍。

1. mysql_connect()函数

打开一个到 MySQL 服务器的连接。如果成功则返回一个 MySQL 连接标识，失败则返回 false。语法如下：

```
resource mysql_connect ( [string server [, string username [, string password [, bool new_link [, int client_flags]]]]] )
```

mysql_connect()函数的参数说明如表 27.3 所示。

表 27.3 mysql_connect()函数的参数说明

参　　数	说　　明
server	MySQL 服务器。可以包括端口号，如"hostname:port"，或者到本地套接字的路径，如对于 localhost 的 ":/path/to/socket"。如果 PHP 指令 mysql.default_host 未定义（默认情况），则默认值是'localhost:3306'
username	用户名。默认值是服务器进程所有者的用户名
password	密码。默认值是空密码
new_link	如果用同样的参数第二次调用 mysql_connect()函数，将不会建立新连接，而将返回已经打开的连接标识。参数 new_link 改变此行为并使 mysql_connect()函数总是打开新的连接，甚至 mysql_connect()函数曾在前面被同样的参数调用过
client_flags	client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_SSL、MYSQL_CLIENT_COMPRESS、MYSQL_CLIENT_IGNORE_SPACE 或 MYSQL_CLIENT_INTERACTIVE

2. mysql_select_db()函数

选择 MySQL 数据库。如果成功返回 true，失败返回 false。语法如下：

```
bool mysql_select_db ( string database_name [, resource link_identifier] )
```

mysql_select_db()函数设定与指定的连接标识符所关联的服务器上的当前激活数据库。如果没有指定连接标识符，则使用上一个打开的连接。如果没有打开连接，本函数将无参数调用 mysql_connect()函数来尝试打开一个并使用。每个其后的 mysql_query()函数调用都会作用于活动数据库。

下面应用 mysql_connect()函数连接 MySQL 服务器，然后应用 mysql_select_db()函数连接 MySQL 数据库。其代码如下：

```
<?php
$conn = mysql_connect("localhost", "root", "111");                          //连接 MySQL 服务器
$db=mysql_select_db("db_pursey", $conn) or die ("数据库连接失败: " .mysql_error()); //连接数据库 db _pursey
mysql_query("set names gb2312");                                            //采用 GB2312 编码方式
?>
```

3. mysql_query()函数

mysql_query()函数用来根据连接标识符向该数据库服务器的当前数据库发送查询。语法格式如下：

```
int mysql_query(string query ,int [link_identifier]);
```

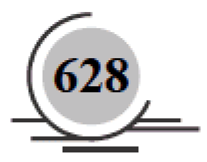
其中，query 是查询字符串；link_identifier 是数据库连接标识符。

mysql_query()函数在执行成功时返回一个结果标识符，失败则返回 false。

4. date()函数

date()函数主要用于格式化一个本地时间/日期。语法如下：

```
string date ( string format , int timestamp)
```



27.6.2 MySQL 数据库连接技术

本模块实现免费供求信息发布功能主要应用到如下几个函数，下面进行详细介绍。

1．mysql_connect()函数

打开一个到 MySQL 服务器的连接。如果成功则返回一个 MySQL 连接标识，失败则返回 false。语法如下：

```
resource mysql_connect ( [string server [, string username [, string password [, bool new_link [, int client_flags]]]]] )
```

mysql_connect()函数的参数说明如表 27.3 所示。

表 27.3 mysql_connect()函数的参数说明

参　数	说　明
server	MySQL 服务器。可以包括端口号，如"hostname:port"，或者到本地套接字的路径，如对于 localhost 的 ":/path/to/socket"。如果 PHP 指令 mysql.default_host 未定义（默认情况），则默认值是'localhost:3306'
username	用户名。默认值是服务器进程所有者的用户名
password	密码。默认值是空密码
new_link	如果用同样的参数第二次调用 mysql_connect()函数，将不会建立新连接，而将返回已经打开的连接标识。参数 new_link 改变此行为并使 mysql_connect()函数总是打开新的连接，甚至 mysql_connect()函数曾在前面被同样的参数调用过
client_flags	client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_SSL、MYSQL_CLIENT_COMPRESS、MYSQL_CLIENT_IGNORE_SPACE 或 MYSQL_CLIENT_INTERACTIVE

2．mysql_select_db()函数

选择 MySQL 数据库。如果成功返回 true，失败返回 false。语法如下：

```
bool mysql_select_db ( string database_name [, resource link_identifier] )
```

mysql_select_db()函数设定与指定的连接标识符所关联的服务器上的当前激活数据库。如果没有指定连接标识符，则使用上一个打开的连接。如果没有打开连接，本函数将无参数调用 mysql_connect()函数来尝试打开一个并使用。每个其后的 mysql_query()函数调用都会作用于活动数据库。

下面应用 mysql_connect()函数连接 MySQL 服务器，然后应用 mysql_select_db()函数连接 MySQL 数据库。其代码如下：

```
<?php
$conn = mysql_connect("localhost", "root", "111");                          //连接 MySQL 服务器
$db=mysql_select_db("db_pursey", $conn) or die ("数据库连接失败: " .mysql_error()); //连接数据库 db _pursey
mysql_query("set names gb2312");                                            //采用 GB2312 编码方式
?>
```

3．mysql_query()函数

mysql_query()函数用来根据连接标识符向该数据库服务器的当前数据库发送查询。语法格式如下：

```
int mysql_query(string query ,int [link_identifier]);
```

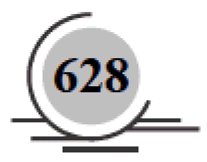
其中，query 是查询字符串；link_identifier 是数据库连接标识符。

mysql_query()函数在执行成功时返回一个结果标识符，失败则返回 false。

4．date()函数

date()函数主要用于格式化一个本地时间/日期。语法如下：

```
string date ( string format , int timestamp)
```



该函数返回将参数 timestamp 按照指定格式格式化而产生的字符串，其中参数 timestamp 是可选的，默认值为 time()，即如果没有给出时间戳则使用本地当前时间。

date()函数的参数 format 的格式化选项如表 27.4 所示。

表 27.4　参数 format 的格式化选项

参　数	说　明
a	小写的上午和下午值，返回值为 am 或 pm
A	大写的上午和下午值，返回值为 AM 或 PM
B	Swatch Internet 标准时，返回值为 000～999
d	月份中的第几天，有前导零的 2 位数字，返回值为 01～31
D	星期中的第几天，文本格式，3 个字母，返回值为 Mon～Sun
F	月份，完整的文本格式，返回值为 January～December
g	小时，12 小时格式，没有前导零，返回值为 1～12
G	小时，24 小时格式，没有前导零，返回值为 0～23
h	小时，12 小时格式，有前导零，返回值为 01～12
H	小时，24 小时格式，有前导零，返回值为 00～23
i	有前导零的分钟数，返回值为 00～59
I	判断是否为夏令时，如果是夏令时返回值为 1，否则为 0
j	月份中的第几天，没有前导零，返回值为 1～31
l	星期数，完整的文本格式，返回值为 Sunday～Saturday
L	判断是否为闰年，如果是闰年返回值为 1，否则为 0
m	数字表示的月份，有前导零，返回值为 01～12
M	3 个字母缩写表示的月份，返回值为 Jan～Dec
n	数字表示的月份，没有前导零，返回值为 1～12
o	与格林威治时间相差的小时数，如 0200
r	RFC 822 格式的日期，如 Thu，21 Dec 2000 16:01:07 +0200
s	秒数，有前导零，返回值为 00～59
S	每月天数后面的英文后缀，2 个字符，如 st、nd、rd 或者 th。可以和 j 一起使用
t	指定月份所应有的天数
T	本机所在的时区
U	从 Unix 纪元（January 1 1970 00:00:00 GMT）开始至今的秒数
w	星期中的第几天，数字表示，返回值为 0～6
W	ISO-8601 格式年份中的第几周，每周从星期一开始
Y	4 位数字完整表示的年份，返回值如 1998、2008
y	2 位数字表示的年份，返回值如 88 或 08
z	年份中的第几天，返回值为 0～366
Z	时差偏移量的秒数。UTC 西边的时区偏移量总是负的，UTC 东边的时区偏移量总是正的，返回值为 -43200～43200

注意　有效的时间戳典型范围是格林威治时间 1901 年 12 月 13 日 20:45:54 到 2038 年 1 月 19 日 03:14:07（此范围符合 32 位有符号整数的最小值和最大值）。在 Windows 系统中此范围限制为从 1970 年 1 月 1 日到 2038 年 1 月 19 日。

例如，应用 date()函数格式化一个日期，并输出日期的值。其代码如下：

```
<?php
echo date("y:m:d");
?>
```

结果为：07:07:11。

27.6.3　免费供求信息发布模块的实现过程

用户通过单击前台页面导航栏的“我要发布”超链接，进入信息发布页面，如图 27.22 所示。程序会先验证用户输入的信息，若验证失败，则返回信息发布页面，进行相应提示。若验证成功，则向数据库中插入记录，完成发布操作。



图 27.22　免费供求信息发布网页

在功能导航页 top.php 中，在“我要发布”图像上添加图像映射，并绘制热区。其代码如下：

（代码位置：光盘\TM\Instance\27\top.php）

```
<map name="Map">                                              //图像映射
    <area shape="rect" coords="356,8,424,28" href="release.php">
</map>
```

在信息发布页面选择要发布的信息类型后，填写真实有效的供求信息。为了避免用户添加空信息，在单击“发布信息”按钮时，应用 JavaScript 脚本自定义一个 checkform()函数，验证提交的表单各元素是否为空值，如果为空，则弹出提示信息，并将焦点定位到为空值的表单元素。其代码如下：

（代码位置：光盘\TM\Instance\27\release_content.php）

```
<script language="javascript">
function checkform(form){                          //自定义一个 JavaScript 函数 checkform()
    for(i=0;i<form.length;i++){                    //应用 for 循环语句检索 form 表单元素的值是否为空
        if(form.elements[i].value==""){            //如果 form 表单中某个元素值为空
            alert("请将发布信息填写完整！");       //弹出提示信息
```

例如，应用 date()函数格式化一个日期，并输出日期的值。其代码如下：

```
<?php
echo date("y:m:d");
?>
```

结果为：07:07:11。

27.6.3　免费供求信息发布模块的实现过程

用户通过单击前台页面导航栏的“我要发布”超链接，进入信息发布页面，如图 27.22 所示。程序会先验证用户输入的信息，若验证失败，则返回信息发布页面，进行相应提示。若验证成功，则向数据库中插入记录，完成发布操作。



图 27.22　免费供求信息发布网页

在功能导航页 top.php 中，在“我要发布”图像上添加图像映射，并绘制热区。其代码如下：

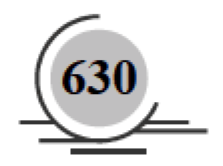
（代码位置：光盘\TM\Instance\27\top.php）

```
<map name="Map">                                        //图像映射
     <area shape="rect" coords="356,8,424,28" href="release.php">
</map>
```

在信息发布页面选择要发布的信息类型后，填写真实有效的供求信息。为了避免用户添加空信息，在单击“发布信息”按钮时，应用 JavaScript 脚本自定义一个 checkform()函数，验证提交的表单各元素是否为空值，如果为空，则弹出提示信息，并将焦点定位到为空值的表单元素。其代码如下：

（代码位置：光盘\TM\Instance\27\release_content.php）

```
<script language="javascript">
function checkform(form){                               //自定义一个 JavaScript 函数 checkform()
     for(i=0;i<form.length;i++){                        //应用 for 循环语句检索 form 表单元素的值是否为空
          if(form.elements[i].value==""){               //如果 form 表单中某个元素值为空
               alert("请将发布信息填写完整！");            //弹出提示信息
```



```
                form.elements[i].focus();                    //将焦点的值定位到为空值的表单元素
                return false;                                //返回焦点
            }
        }
}
</script>
```

创建与数据库 db_pursey 的连接，代码如下：

（代码位置：光盘\TM\Instance\27\conn\conn.php）

```
<?php
$link=mysql_connect("localhost","root","111");               //连接 MySQL 服务器
mysql_select_db("db_pursey ",$link);                          //连接 MySQL 数据库文件 db_pursey
mysql_query("set names gb2312");                              //采用 GB2312 编码方式
?>
```

提交表单信息到数据处理页，应用 insert…into 语句向免费供求信息表中添加供求信息。如果信息添加成功，则弹出成功的信息提示；否则弹出失败的提示信息。其代码如下：

（代码位置：光盘\TM\Instance\27\release_ok.php）

```
<?php
include("conn/conn.php");                                     //连接数据库文件
$type=$_POST[type];                                           //获取信息类型
$title=$_POST[title];                                         //获取信息标题
$content=$_POST[content];                                     //获取信息内容
$linkman=$_POST[linkman];                                     //获取联系人
$tel=$_POST[tel];                                             //获取联系电话
❶     $edate=date("Y-m-d h:i:s");                             //获取发布时间
❷     $sql=mysql_query("insert into tb_info(type,title,content,linkman,tel,checkstate,edate)
values('$type','$title','$content','$linkman','$tel',0,'$edate')");   //将免费的供求信息添加到数据表中
if($sql){                                                     //如果添加操作成功，则弹出提示信息
      echo "<script>alert('恭喜您，信息发布成功！');window.location.href='release.php';</script>";
}else{                                                        //如果添加操作失败，则弹出提示信息
      echo "<script>alert('对不起，信息发布失败！');history.back();</script>";
}
?>
```

说明

❶ date：格式化一个本地时间/日期。

❷ insert…into：向指定的数据表中添加数据信息。

27.7　信息检索模块设计

视频讲解：光盘\TM\Video\第 27 章\信息检索模块设计.exe

27.7.1　信息检索模块概述

信息检索是对已存在于数据库中的数据按条件进行筛选浏览，是查看历史信息和确认数据操作最为快速、有效的办法。信息检索模块主要通过选择信息类型和输出查询关键字模糊查询供求信息资源，并输出查询结果。考虑到供求信息的信息量较大，因此本模块对与查询关键字相匹配的查询结果进行描红，从而方便用户的浏览。信息检索模块的示意图如图 27.23 所示。

```
                    form.elements[i].focus();                              //将焦点的值定位到为空值的表单元素
                    return false;                                          //返回焦点
                }
            }
}
</script>
```

创建与数据库 db_pursey 的连接，代码如下：

（代码位置：光盘\TM\Instance\27\conn\conn.php）

```
<?php
$link=mysql_connect("localhost","root","111");                //连接 MySQL 服务器
mysql_select_db("db_pursey ",$link);                          //连接 MySQL 数据库文件 db_pursey
mysql_query("set names gb2312");                              //采用 GB2312 编码方式
?>
```

提交表单信息到数据处理页，应用 insert…into 语句向免费供求信息表中添加供求信息。如果信息添加成功，则弹出成功的信息提示；否则弹出失败的提示信息。其代码如下：

（代码位置：光盘\TM\Instance\27\release_ok.php）

```
<?php
include("conn/conn.php");                                     //连接数据库文件
$type=$_POST[type];                                           //获取信息类型
$title=$_POST[title];                                         //获取信息标题
$content=$_POST[content];                                     //获取信息内容
$linkman=$_POST[linkman];                                     //获取联系人
$tel=$_POST[tel];                                             //获取联系电话
❶    $edate=date("Y-m-d h:i:s");                              //获取发布时间
❷    $sql=mysql_query("insert into tb_info(type,title,content,linkman,tel,checkstate,edate)
values('$type','$title','$content','$linkman','$tel',0,'$edate')");    //将免费的供求信息添加到数据表中
if($sql){                                                     //如果添加操作成功，则弹出提示信息
    echo "<script>alert('恭喜您，信息发布成功！');window.location.href='release.php';</script>";
}else{                                                        //如果添加操作失败，则弹出提示信息
    echo "<script>alert('对不起，信息发布失败！');history.back();</script>";
}
?>
```

说明

❶ date：格式化一个本地时间/日期。

❷ insert…into：向指定的数据表中添加数据信息。

27.7　信息检索模块设计

视频讲解：光盘\TM\Video\第 27 章\信息检索模块设计.exe

27.7.1　信息检索模块概述

信息检索是对已存在于数据库中的数据按条件进行筛选浏览，是查看历史信息和确认数据操作最为快速、有效的办法。信息检索模块主要通过选择信息类型和输出查询关键字模糊查询供求信息资源，并输出查询结果。考虑到供求信息的信息量较大，因此本模块对与查询关键字相匹配的查询结果进行描红，从而方便用户的浏览。信息检索模块的示意图如图 27.23 所示。

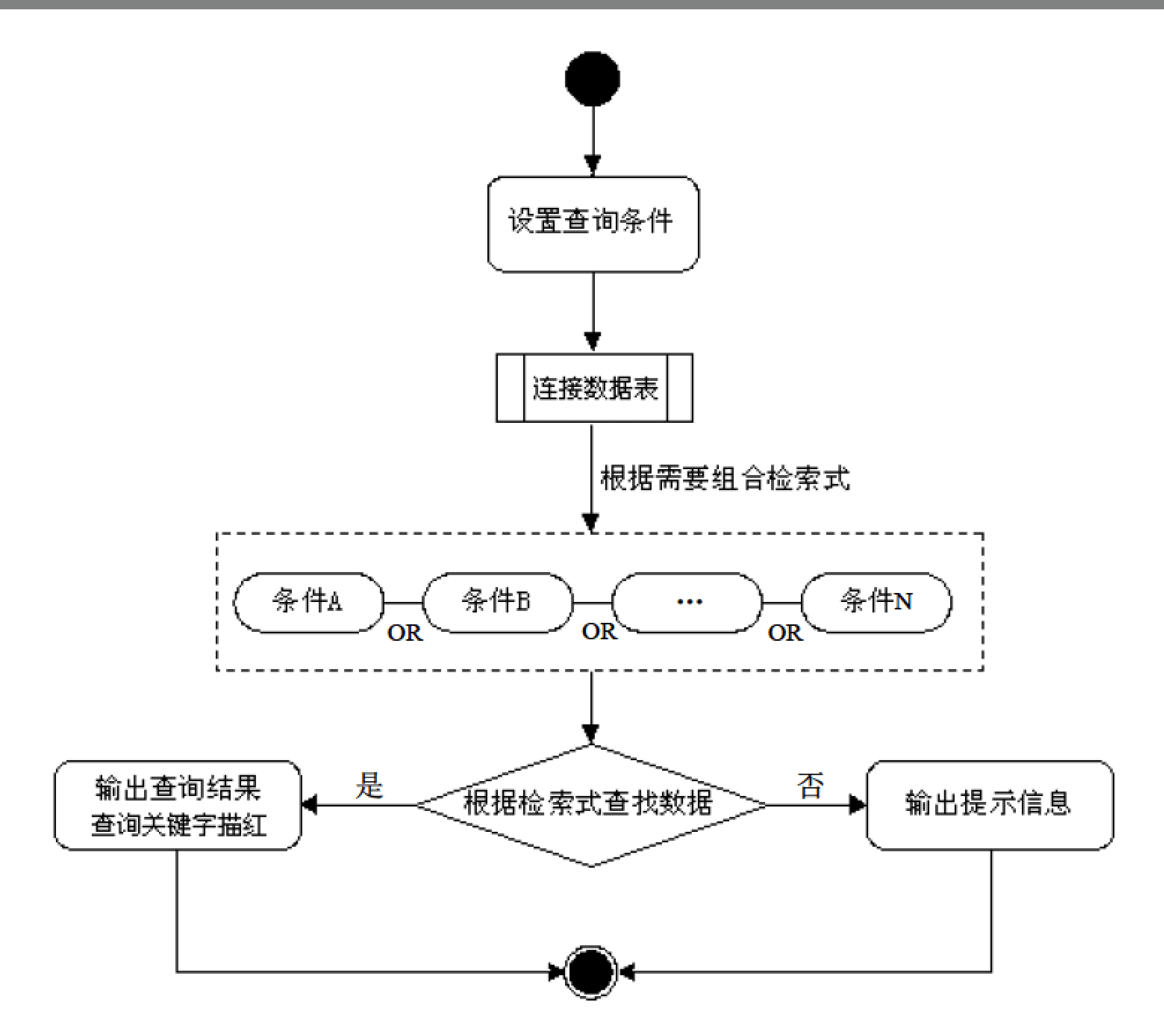


图 27.23　信息检索模块的示意图

27.7.2　模糊查询技术

在对数据进行查询后，最终需要将查询结果显示在页面中反馈给浏览者。在 PHP 中，查询结果的显示方式有很多种，最常用的就是表格显示方式。因为采用这种方式显示的数据条理清晰、简捷明了。

在利用表格显示查询结果时，通常是将查询结果保存在结果集中，然后需要使用 do…while 循环将其查询结果显示出来。需要注意的是，需要先判断查询结果是否为空，只有查询结果不为空时，才可以使用循环语句显示数据。为了使读者更好地理解，下面给出其实现流程图，如图 27.24 所示。

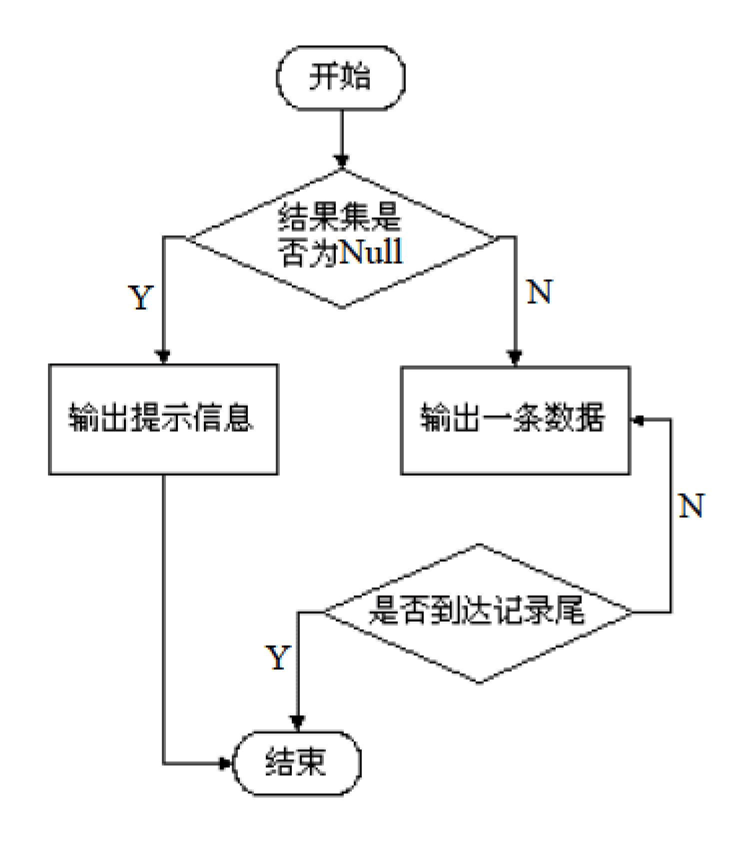
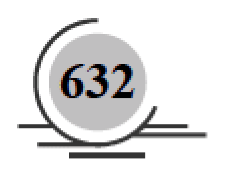


图 27.24　信息检索模块流程图

考虑到用户不可能全面了解数据表中的数据信息，如不能确定所要查询信息的内容、查询的主题等，这时就需要使用 like 进行模糊查询。like 关键字需要使用通配符在字符串内查找指定的模式，所以读者需要了解通配符及其含义。通配符的含义如表 27.5 所示。

表 27.5　like 关键字中的通配符及说明

通　配　符	说　　明
%	由零个或更多字符组成的任意字符串
_	任意单个字符
[ ]	用于指定范围，如[A～F]，表示 A～F 内的任何单个字符
[^]	表示指定范围之外的，如[^ A～F]表示 A～F 以外的任何单个字符



如果想查询包含“女子公寓”的信息，可以使用 like 运算符配合通配符“%”完成。其 SQL 语句如下：

```
select * from tb_info where content like '%女子公寓%';
```

如果想查找信息类型为“公寓信息”或者内容为“女子公寓”的信息时配合 or 运算符来使用。其 SQL 语句如下：

```
select * from tb_info  where type='公寓信息' or content like'%女子公寓%'
```

本模块实现付费信息与查询关键字相匹配的信息的 SQL 语句如下：

```
$type=$_POST[type];                                                    //获取信息类型
$content=$_POST[content];                                              //获取查询关键字
$sql1=mysql_query("select * from tb_leaguerinfo  where checkstate=1 and type='$type' and content
like'%$content%' or title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info1=mysql_fetch_array($sql1);                                       //检索付费的信息
```

本模块实现免费信息与查询关键字相匹配的信息的 SQL 语句如下：

```
$sql=mysql_query("select * from tb_info  where checkstate=1 and type='$type' and content like'%$content%' or
title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info=mysql_fetch_array($sql);                                         //检索免费的信息
```

注意 当满足数据表中多个字段中的任一字段时，可以使用 or 运算符将多个条件连接。

另外，由于搜索的内容中文字比较多，为了方便浏览者查找自己所关注的内容信息，所以在搜索引擎中加入了描红功能。描红功能主要用 str_ireplace()函数实现。

27.7.3　信息检索模块的实现过程

在开发信息检索模块时，由于该网站含有大量的数据信息，为了方便用户浏览网站信息，需要添加复合条件查询实现搜索功能。在信息检索区的“关键字”文本框中输入欲查询的关键字，在“条件”下拉列表框中选择要搜索的信息类型，然后单击“开始搜索”按钮，对指定条件的记录进行检索并输出结果集到浏览器，同时，为了方便浏览者查找自己所关注的内容信息，本模块对查询关键字进行了描红。运行结果如图 27.25 所示。



图 27.25　信息检索页面的运行结果

如果想查询包含“女子公寓”的信息，可以使用 like 运算符配合通配符“%”完成。其 SQL 语句如下：

```
select * from tb_info where content like '%女子公寓%';
```

如果想查找信息类型为“公寓信息”或者内容为“女子公寓”的信息时配合 or 运算符来使用。其 SQL 语句如下：

```
select * from tb_info   where type='公寓信息' or content like'%女子公寓%'
```

本模块实现付费信息与查询关键字相匹配的信息的 SQL 语句如下：

```
$type=$_POST[type];                                                          //获取信息类型
$content=$_POST[content];                                                    //获取查询关键字
$sql1=mysql_query("select * from tb_leaguerinfo  where checkstate=1 and type='$type' and content like'%$content%' or title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info1=mysql_fetch_array($sql1);                                             //检索付费的信息
```

本模块实现免费信息与查询关键字相匹配的信息的 SQL 语句如下：

```
$sql=mysql_query("select * from tb_info  where checkstate=1 and type='$type' and content like'%$content%' or title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info=mysql_fetch_array($sql);                                               //检索免费的信息
```

注意　当满足数据表中多个字段中的任一字段时，可以使用 or 运算符将多个条件连接。

另外，由于搜索的内容中文字比较多，为了方便浏览者查找自己所关注的内容信息，所以在搜索引擎中加入了描红功能。描红功能主要用 str_ireplace()函数实现。

27.7.3　信息检索模块的实现过程

在开发信息检索模块时，由于该网站含有大量的数据信息，为了方便用户浏览网站信息，需要添加复合条件查询实现搜索功能。在信息检索区的“关键字”文本框中输入欲查询的关键字，在“条件”下拉列表框中选择要搜索的信息类型，然后单击“开始搜索”按钮，对指定条件的记录进行检索并输出结果集到浏览器，同时，为了方便浏览者查找自己所关注的内容信息，本模块对查询关键字进行了描红。运行结果如图 27.25 所示。



图 27.25　信息检索页面的运行结果

信息检索页面中所涉及到的重要表单元素如表 27.6 所示。

表 27.6　信息检索页面所涉及的重要表单元素

名　　称	元 素 类 型	重 要 属 性	含　　义
form1	form	method="post" action="findinfo.php"	表单
content	text	id="content" size="20"	查询关键字
type	select	<select name="type"> <option value="招聘信息">-招聘信息-</option> <option value="求职信息" selected>-求职信息-</option> </select>	信息类型
search	image	src="Images/btn1.gif" onClick="return chkinput(form)"	开始搜索按钮

应用 JavaScript 脚本自定义一个 chkinput()函数，实现对表单提交的信息进行验证。其代码如下：

（代码位置：光盘\TM\Instance\27\left.php）

```
<script language="javascript">
        function chkinput(form){                                    //自定义一个 chkinput()函数
          if(form.content.value==""){                               //判断如果查询关键字文本框等于空
             alert("请输入查询关键字!");                              //则弹出提示信息
             form.content.select();                                 //重新定位焦点
             return false;                                          //返回表单元素
          }
        }
</script>
```

将表单信息提交到数据处理页，连接数据库文件，接收表单信息，然后用 mysql_query()函数向服务器发送 SQL 语句，检索与查询关键字相匹配的信息资源。其代码如下：

（代码位置：光盘\TM\Instance\27\findinfo.php）

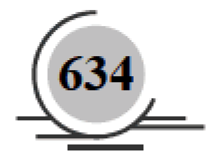
```
<?php
include("conn/conn.php");                                           //连接数据库文件
$type=$_POST[type];                                                 //获取信息类型
$content=$_POST[content];                                           //获取查询关键字
$sql1=mysql_query("select * from tb_leaguerinfo  where checkstate=1 and type='$type' and content
like'%$content%' or title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info1=mysql_fetch_array($sql1);                                    //检索付费的供求信息
$sql=mysql_query("select * from tb_info  where checkstate=1 and type='$type' and content like'%$content%' or
title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info=mysql_fetch_array($sql);                                      //检索免费的供求信息
?>
```

说明 信息检索需要从免费信息表 tb_info 和付费信息表 tb_leaguerinfo 中获取数据，因此需要向 MySQL 服务器传递两条 SQL 语句。

用 do…while 循环语句输出付费信息与查询关键字相匹配的信息资源，并用 str_ireplace 函数对查询关键字实现描红功能。其代码如下：

（代码位置：光盘\TM\Instance\27\findinfo.php）

```
<!--下面输出的是付费信息与查询关键字相匹配的信息    -->
<?php
If($info1){                                                         //如果检索到了付费信息
```



信息检索页面中所涉及到的重要表单元素如表 27.6 所示。

表 27.6　信息检索页面所涉及的重要表单元素

名　称	元素类型	重要属性	含　义
form1	form	method="post" action="findinfo.php"	表单
content	text	id="content" size="20"	查询关键字
type	select	<select name="type"> <option value="招聘信息">-招聘信息-</option> <option value="求职信息" selected>-求职信息-</option> </select>	信息类型
search	image	src="Images/btn1.gif" onClick="return chkinput(form)"	开始搜索按钮

应用 JavaScript 脚本自定义一个 chkinput()函数，实现对表单提交的信息进行验证。其代码如下：

（代码位置：光盘\TM\Instance\27\left.php）

```
<script language="javascript">
        function chkinput(form){                                    //自定义一个 chkinput()函数
          if(form.content.value==""){                               //判断如果查询关键字文本框等于空
             alert("请输入查询关键字!");                              //则弹出提示信息
             form.content.select();                                 //重新定位焦点
             return false;                                          //返回表单元素
          }
        }
</script>
```

将表单信息提交到数据处理页，连接数据库文件，接收表单信息，然后用 mysql_query()函数向服务器发送 SQL 语句，检索与查询关键字相匹配的信息资源。其代码如下：

（代码位置：光盘\TM\Instance\27\findinfo.php）

```
<?php
include("conn/conn.php");                                           //连接数据库文件
$type=$_POST[type];                                                 //获取信息类型
$content=$_POST[content];                                           //获取查询关键字
$sql1=mysql_query("select * from tb_leaguerinfo  where checkstate=1 and type='$type' and content
like'%$content%' or title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info1=mysql_fetch_array($sql1);                                    //检索付费的供求信息
$sql=mysql_query("select * from tb_info  where checkstate=1 and type='$type' and content like'%$content%' or
title like'%$content%' or linkman like'%$content%' or tel like'%$content%'");
$info=mysql_fetch_array($sql);                                      //检索免费的供求信息
?>
```

说明　信息检索需要从免费信息表 tb_info 和付费信息表 tb_leaguerinfo 中获取数据，因此需要向 MySQL 服务器传递两条 SQL 语句。

用 do…while 循环语句输出付费信息与查询关键字相匹配的信息资源，并用 str_ireplace 函数对查询关键字实现描红功能。其代码如下：

（代码位置：光盘\TM\Instance\27\findinfo.php）

```
<!--下面输出的是付费信息与查询关键字相匹配的信息   -->
<?php
If($info1){                                                         //如果检索到了付费信息
```

```
    do{                                              //则用 do…while 循环语句输出付费信息
 ?>
    <table width="540" border="0" cellspacing="0" cellpadding="0">
      <tr>
        <td height="26">
          <!-- 应用 str_ireplace)函数对查询关键字进行描红 -->
          <!-- 对与查询关键字所匹配的信息类型进行描红 -->
          『<?php echo str_ireplace($content,"<font color='#FF0000'>".$content."</font>",$info1 [type]);?>』
nbsp;
          <!-- 对与查询关键字所匹配的信息标题进行描红 -->
          <?php echo str_ireplace($content,"<font color='#FF0000'>".$content."</font>",$info1 [title]);?> 
 
          <!-- 对与查询关键字所匹配的发布时间进行描红 -->
          <?php echo str_ireplace($content,"<font color='#FF0000'>".$content."</font>",$info1 [edate]);?></td>
        </tr>
        <tr>
          <td height="26">    
          <!-- 对与查询关键字所匹配的信息内容进行描红 -->
          <?php echo str_ireplace($content,"<font color='#FF0000'>".$content."</font>",$info1 [content]);?>
</td>
        </tr>
        <tr>
          <td height="26"> 联系人：
          <!-- 对与查询关键字所匹配的联系人进行描红 -->
          <?php echo str_ireplace($content,"<font color='#FF0000'>".$content."</font>",$info1 [linkman]);?>
 
          <!-- 对与查询关键字所匹配的联系电话进行描红 -->
          联系电话：<?php echo str_ireplace($content,"<font color='#FF0000'>".$content."</font>",$info1
[tel]);?>
          <!-- ------------------------------------------------------------ -->
          </td>
    </tr>
  </table>
<?php
    }while($info1=mysql_fetch_array($sql1));          //循环语句结束
?>
```

说明 免费信息的输出方式与付费信息的基本类似，代码部分略，详见本书附赠光盘。

免费信息的输出方式与付费信息的基本类似，下面给出实现过程的核心代码结构：

（代码位置：光盘\TM\Instance\27\findinfo.php）

```
<!-- ---------------------------------下面输出的是免费信息与查询关键字相匹配的信息------------------------------ -->
<?php
if($info){                                           //如果检索到了免费信息
    do{                                              //则用 do…while 循环语句输出付费信息
?>
    …                                                //免费信息的输出方式与付费信息的类似，代码略
<?php
    } while($info=mysql_fetch_array($sql));          //do…while 循环语句结束
    }                                                //if 条件语句结束
?>
```

如果在免费信息表和付费信息表中没有检索到与查询关键字相匹配的数据，则弹出提示信息。其代码

如下：

（代码位置：光盘\TM\Instance\27\findinfo.php）

```
<table width="540" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td align="center">您检索的信息资源不存在！</td>
    </tr>
</table>
```

27.8　后台首页设计

视频讲解：光盘\TM\Video\第 27 章\后台首页设计.exe

27.8.1　后台首页概述

程序开发人员在设计网站后台主页时，主要从后台管理人员对功能的易操作实用性、网站的易维护性考虑，因此采用了框架技术。易查供求信息网后台首页主要包含以下内容。

- ☑ 发布付费的供求信息（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等），以及付费信息的浏览、审核及删除功能。
- ☑ 免费信息的浏览、审核及删除功能。
- ☑ 企业广告信息的发布、浏览、前台推荐显示、删除功能。
- ☑ 网站首页：为管理员进入前台提供一个入口。
- ☑ 退出登录：注销当前用户。

下面看一下本案例中提供的后台首页，该页面在本书光盘中的路径为 TM\27\admin\index.php，如图 27.26 所示。

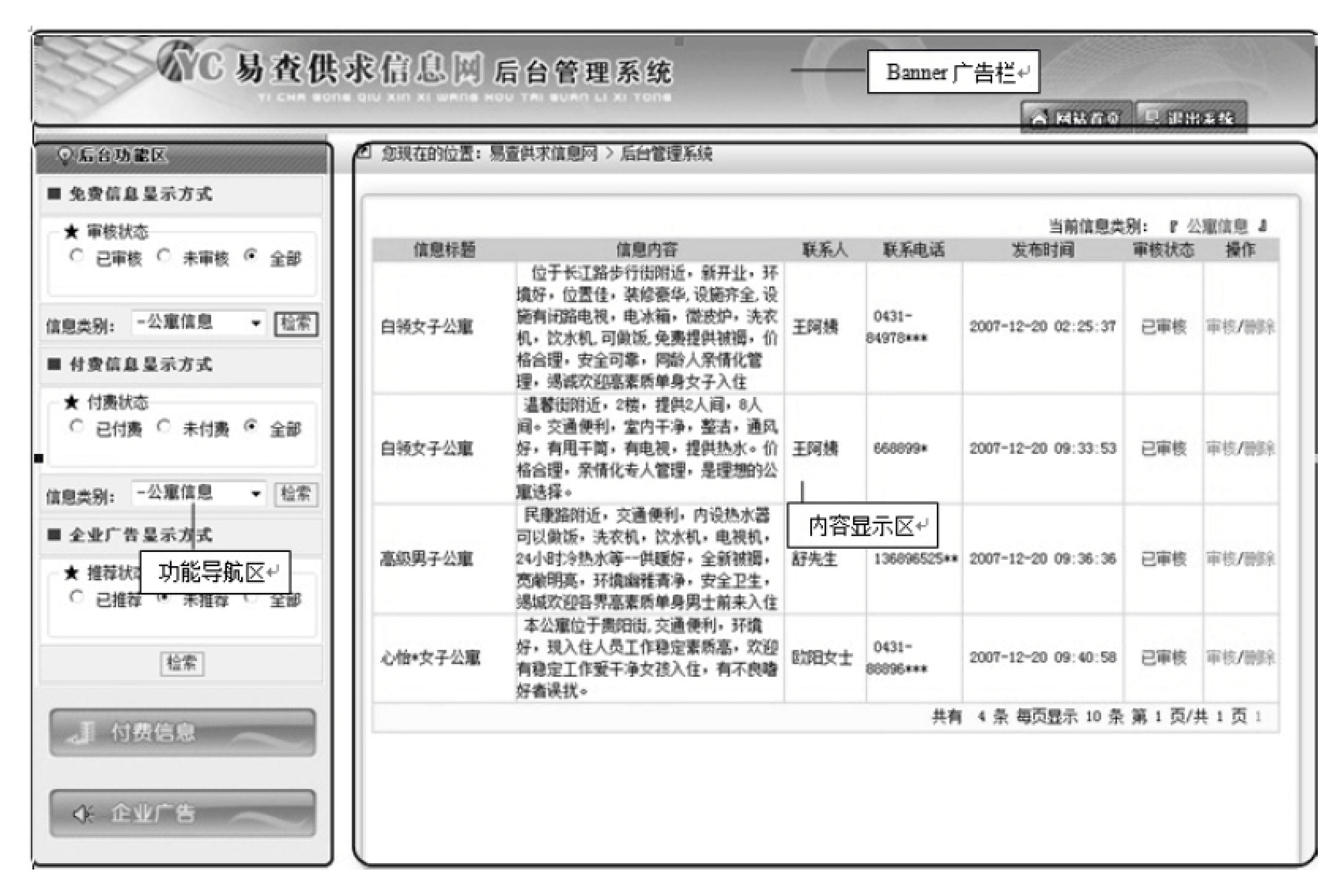


图 27.26　供求信息网站后台首页



如下：

（代码位置：光盘\TM\Instance\27\findinfo.php）

```
<table width="540" border="0" cellspacing="0" cellpadding="0">
    <tr>
      <td align="center">您检索的信息资源不存在！ </td>
    </tr>
</table>
```

27.8　后台首页设计

视频讲解：光盘\TM\Video\第 27 章\后台首页设计.exe

27.8.1　后台首页概述

程序开发人员在设计网站后台主页时，主要从后台管理人员对功能的易操作实用性、网站的易维护性考虑，因此采用了框架技术。易查供求信息网后台首页主要包含以下内容。

☑ 发布付费的供求信息（包括公寓信息、招聘信息、求职信息、培训信息、家教信息、房屋信息、车辆信息、求购信息、出售信息、招商引资、寻人/物启示等），以及付费信息的浏览、审核及删除功能。

☑ 免费信息的浏览、审核及删除功能。

☑ 企业广告信息的发布、浏览、前台推荐显示、删除功能。

☑ 网站首页：为管理员进入前台提供一个入口。

☑ 退出登录：注销当前用户。

下面看一下本案例中提供的后台首页，该页面在本书光盘中的路径为 TM\27\admin\index.php，如图 27.26 所示。

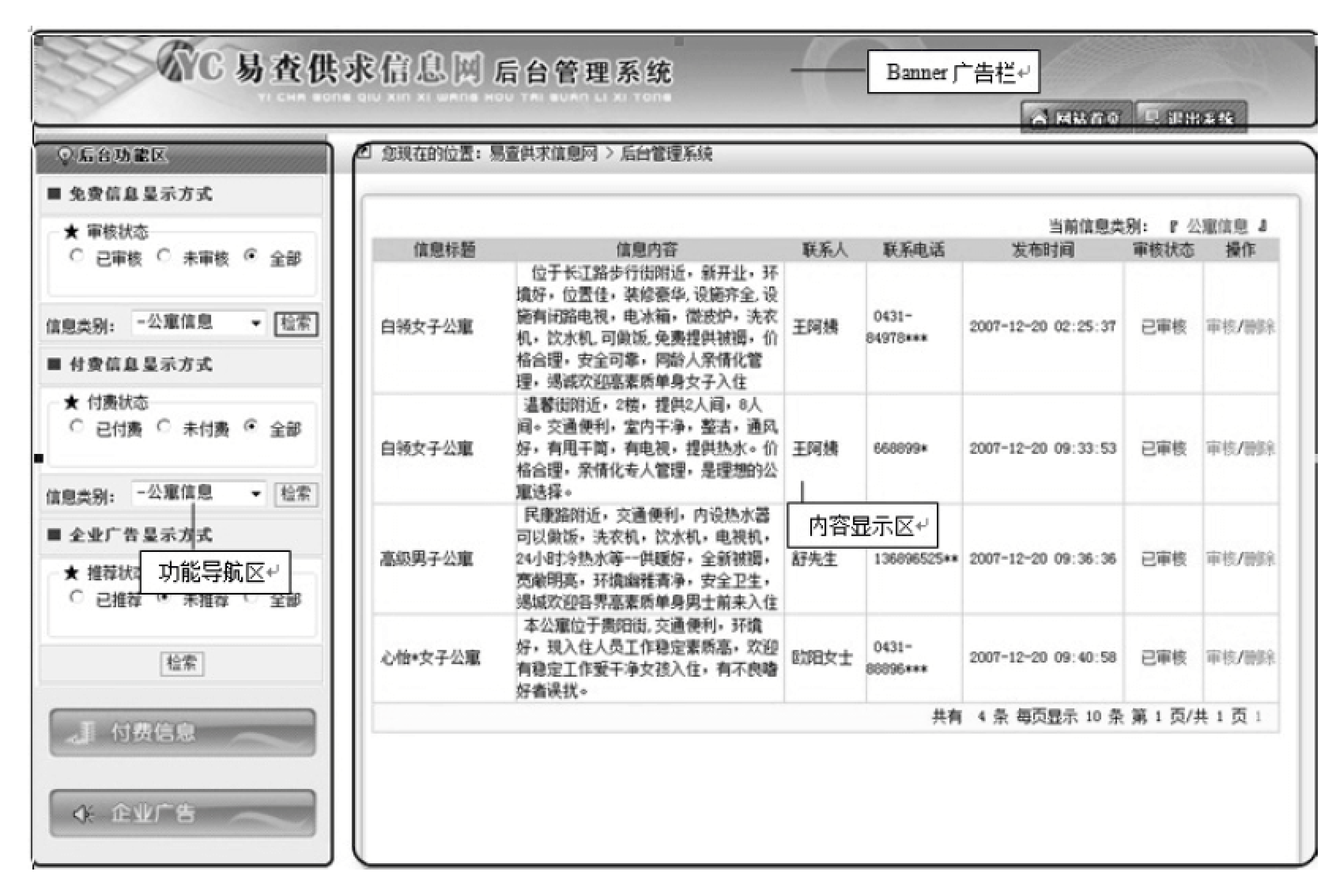


图 27.26　供求信息网站后台首页

27.8.2　frame 框架技术

易查供求信息网后台采用框架技术进行页面布局。所谓框架就是网页的各部分为相互独立的网页，又由一个网页将这些分开的网页组成一个完整的网页，显示在浏览者的浏览器中，重复出现的内容被固定下来，每次浏览者发出对页面的请求时，只下载发生变化的框架页面，其他子页面保持不变。

使用框架可以将容器窗口划分为若干个子窗口，在每个子窗口可以分别显示不同的网页。首先在 Dreamweaver 8 中创建一个“左→中→右”的框架集，然后在标识①中添加“上→中→下”的框架集，最后在标识②中添加一个“左→右”的框架集，从而完成一个完整的后台框架。构建框架的流程如图 27.27 所示。

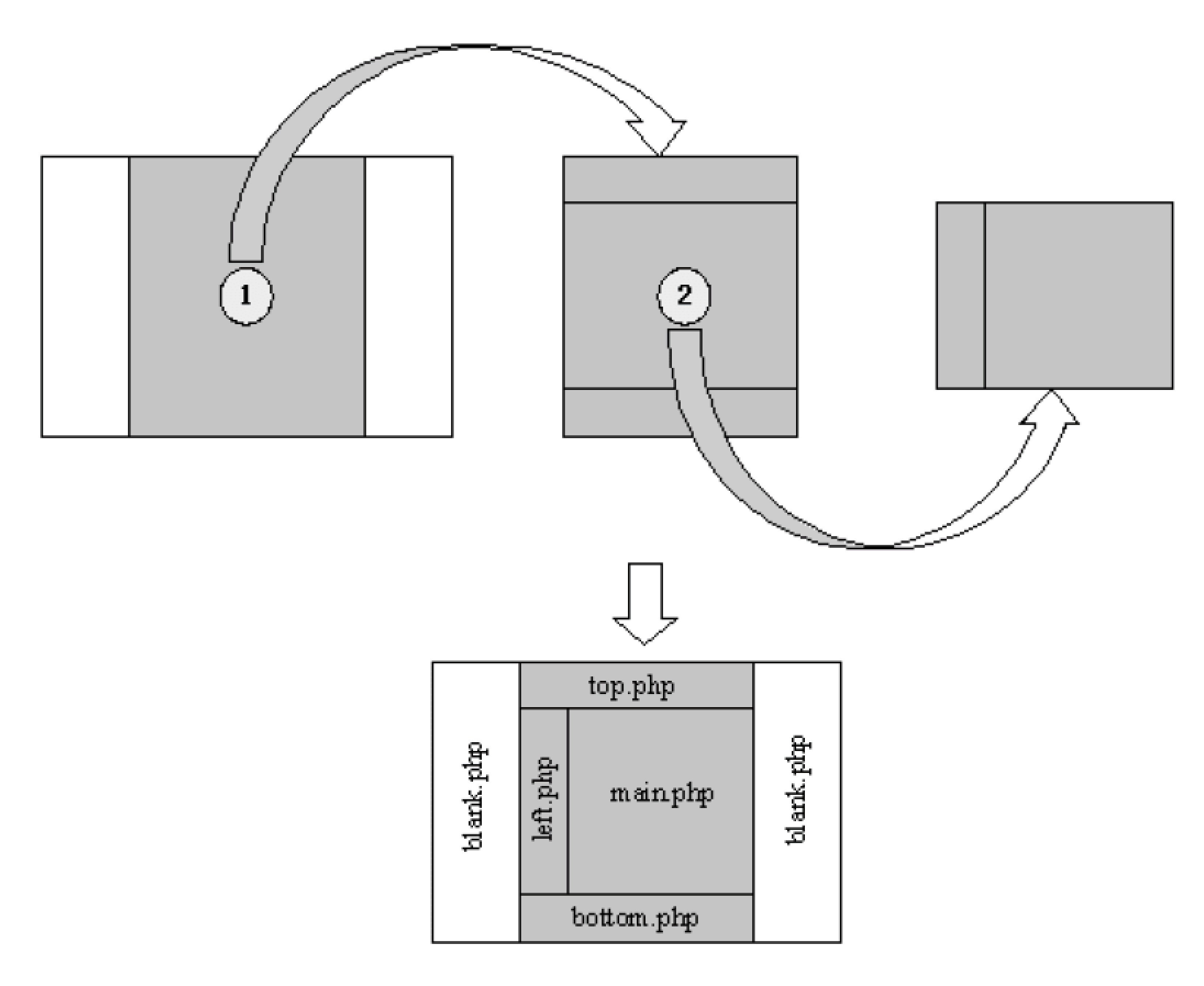


图 27.27　网站后台框架流程

使用框架可以非常方便地完成导航工作。下面详细介绍框架的基本结构、设置框架集属性和框架属性。

1．框架网页的基本结构

框架网页通过一个或多个 FRAMESET 和 FRAME 标记来定义。在框架网页中，将 FRAMESET 标记置于 HEAD 标记后，以取代 BODY 标记的位置，当客户端浏览器不支持框架网页时，还可以使用 NOFRAMES 标记给出框架不能被显示时的替换内容。框架网页的基本结构可以表示如下：

```
<html>
<head>
<title>基本框架页</title>
</head>
<frameset>
    <frame>
    <frame>
</frameset>
<noframes>
```

```
		<body>
		对不起！您的浏览器不支持框架页面的显示！
		</body>
</noframes>
</html>
```

2. 设置框架集的属性

框架集包含如何组织各个框架的信息，可以通过 FRAMESET 标记来定义。框架是按照行和列来组织的，可以使用 FRAMESET 标记的下列属性对框架的结构进行设置。

（1）左右分割窗口属性 COLS

在水平方向上将浏览器分割成多个窗口，可以通过框架的左右分割窗口属性 COLS 实现。其语法格式如下：

```
<frameset cols="value,value,…">
  <frame>
  <frame>
</frameset>
```

参数 value 用于指定各个框架的列宽，取值有像素、百分比（%）和相对尺寸（*）3 种形式。

例如，若要通过框架将浏览器窗口划分为 3 列，其中第 1 列占浏览器窗口宽度的 20%，第 2 列为 120 像素，第 3 列为浏览器窗口的剩余部分。其代码如下：

```
<frameset cols="20%,120,*" >
  <frame>
  <frame>
</frameset>
```

技巧 如果将 COLS 属性设置为“*，*，*”，则表示将窗口划分成 3 个等宽的框架；如果将 COLS 属性设置为“*，2*，3*”，则表示左边的框架占窗口宽度的 1/6，中间的框架占窗口宽度的 1/3，右边的框架占窗口宽度的 1/2。

（2）上下分割窗口属性 ROWS

在垂直方向上将浏览器分割成多个窗口，可以通过框架的上下分割窗口属性 ROWS 实现。其语法格式如下：

```
<frameset rows="value,value,…">
  <frame>
  <frame>
</frameset>
```

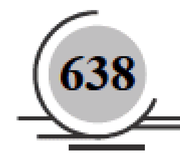
value 用于指定各个框架的行高，取值有像素、百分比（%）和相对尺寸（*）3 种形式，设置方法与 COLS 属性类似。

例如，若要通过框架将浏览器窗口划分为 3 行，其中的第 1 行占浏览器窗口宽度的 20%，第 2 行为 120 像素，第 3 行为浏览器窗口的剩余部分的框架。其代码如下：

```
<frameset rows="20%,120,*" >
  <frame>
  <frame>
</frameset>
```

（3）框架边框显示属性 FRAMEBORDER

该属性用于指定框架周围是否显示边框，取值为 1（显示边框，默认值）或 0（不显示边框）。



```
        <body>
        对不起！您的浏览器不支持框架页面的显示！
        </body>
</noframes>
</html>
```

2. 设置框架集的属性

框架集包含如何组织各个框架的信息，可以通过 FRAMESET 标记来定义。框架是按照行和列来组织的，可以使用 FRAMESET 标记的下列属性对框架的结构进行设置。

（1）左右分割窗口属性 COLS

在水平方向上将浏览器分割成多个窗口，可以通过框架的左右分割窗口属性 COLS 实现。其语法格式如下：

```
<frameset cols="value,value,…">
  <frame>
  <frame>
</frameset>
```

参数 value 用于指定各个框架的列宽，取值有像素、百分比（%）和相对尺寸（*）3 种形式。

例如，若要通过框架将浏览器窗口划分为 3 列，其中第 1 列占浏览器窗口宽度的 20%，第 2 列为 120 像素，第 3 列为浏览器窗口的剩余部分。其代码如下：

```
<frameset cols="20%,120,*" >
  <frame>
  <frame>
</frameset>
```

技巧 如果将 COLS 属性设置为“*，*，*”，则表示将窗口划分成 3 个等宽的框架；如果将 COLS 属性设置为“*，2*，3*”，则表示左边的框架占窗口宽度的 1/6，中间的框架占窗口宽度的 1/3，右边的框架占窗口宽度的 1/2。

（2）上下分割窗口属性 ROWS

在垂直方向上将浏览器分割成多个窗口，可以通过框架的上下分割窗口属性 ROWS 实现。其语法格式如下：

```
<frameset rows="value,value,…">
  <frame>
  <frame>
</frameset>
```

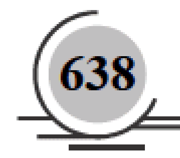
value 用于指定各个框架的行高，取值有像素、百分比（%）和相对尺寸（*）3 种形式，设置方法与 COLS 属性类似。

例如，若要通过框架将浏览器窗口划分为 3 行，其中的第 1 行占浏览器窗口宽度的 20%，第 2 行为 120 像素，第 3 行为浏览器窗口的剩余部分的框架。其代码如下：

```
<frameset rows="20%,120,*" >
  <frame>
  <frame>
</frameset>
```

（3）框架边框显示属性 FRAMEBORDER

该属性用于指定框架周围是否显示边框，取值为 1（显示边框，默认值）或 0（不显示边框）。



（4）FRAMESPACING

该属性用于指定框架之间的间隔，以像素为单位。如果不设置该属性，则框架之间没有间隔。

（5）指定边框宽度属性 BORDER

该属性用于指定边框的宽度，只有 FRAMEBORDER 属性为 1 时有效。

3. 设置框架的属性

使用 FRAME 标记可以设置框架的属性，包括框架的名称、框架是否包含滚动条以及在框架中显示的网页等。其语法格式如下：

```
<frame name="框架名称" src="文件" frameborder="数值" scrolling="值" [noresize] >
```

属性说明如下。

- ☑ name：指定框架的名称。
- ☑ src：指定在框架中显示的网页文件（包括 HTML、ASP 等网页文件）。
- ☑ frameborder：指定框架周围是否显示边框，取值为 1（显示）或 0（不显示）。默认值为 1。
- ☑ scrolling：指定框架是否包含滚动条。如果将该属性设置为 yes，则框架包含滚动条；若将该属性设置为 no，则框架不包含滚动条；如果将该属性设置为 auto，则在需要时包含滚动条。
- ☑ noresize：可选属性，若指定了该属性，则不能调整框架的大小。

27.8.3　后台首页的实现过程

根据以上两节的页面概述及实现技术分析，需要分别创建实现各区域的 PHP 文件，如实现 Banner 广告栏的 top.php、功能导航栏的 left.php、内容显示区的 main.php 和页尾文件 bottom.php 等。下面给出实现该系统后台框架布局的完整代码。

（代码位置：光盘\TM\Instance\27\admin\index.php）

```
<frameset rows="*" cols="1*,16005,1*" framespacing="0" frameborder="NO" border="0">
  <frame src="blank.php" name="left" scrolling="NO" noresize>                    <!--设置空框架页-->
  <frameset rows="1005,*" cols="*" framespacing="0" frameborder="NO" border="0">
          <frameset rows="94,*,190" cols="*" framespacing="0" frameborder="NO" border="0">
            <frame src="top.php" name="topFrame" scrolling="NO" noresize>
                <frameset   rows="*" cols="229,*"   framespacing="0" frameborder="NO" border="0">
                    <frame src="left.php" name="leftFrame" scrolling="NO" noresize>
                    <frame src="main.php" name="mainFrame" scrolling="NO" noresize>
               </frameset>
            <frame src="bottom.php" name="bottomFrame" scrolling="NO" noresize>
          </frameset>
          <frame src="blank.php" name="right" scrolling="NO" noresize>
     </frameset>
<frame src="blank.php"></frameset>                                              <!--设置空框架页-->
<noframes><body>
</body></noframes>
```

注意 在建设 Web 网站时，如何让不同分辨率的用户都能看到网页的最佳效果是程序员在设计之初所要考虑的首要问题。为了使屏蔽的分辨率在大于 1024×768 像素的设置时仍然处于居中显示，只需要在设置框架布局时，在主框架两侧各设置一个宽度相同的 blank.php 空页即可。

27.9　付费供求信息发布模块设计

视频讲解：光盘\TM\Video\第 27 章\付费供求信息发布模块设计.exe

27.9.1　付费供求信息发布模块概述

付费供求信息发布是供求信息网站非常重要的功能，也是供求信息网站的盈利点。企业或用户可以根据自身需要对供求信息先进行付费，付费后由管理员在后台将供求信息发布到相应的信息类别中（共包括 11 个信息类别：招聘信息、求职信息、培训信息、公寓信息、家教信息、车辆信息、物品求购、物品出售、求兑出兑、寻求合作、企业广告）。供求信息成功发布后，管理员需要在后台对发布的供求信息进行审核，如果审核通过后，则显示在前台相应的信息类别网页中。付费供求信息发布的流程如图 27.28 所示。

管理员
发布的信息
管理页面
审核
N
Y
在前台展示发布的信息

图 27.28　付费供求信息发布流程图

27.9.2　计算供求信息的有效时间

付费供求信息与免费供求信息不同的是，付费供求信息不仅需要收取一定的费用，而且还需要一定的时间限制，例如，网站要求一个月（按 30 天计算）每条信息交 10 元的信息费，如果用户交纳 20 元，那么信息显示的天数就是 60 天。在前台进行显示时，不需要管理员进行手动管理，而是通过程序直接计算出信息显示的截止时间。

信息显示的截止时间=系统当前日期+信息的有效天数（与用户交纳的信息费相关）。

自动计算信息显示的截止时间的具体代码如下：

```
$days=$_POST[days];                                    //通过表单传值获取信息显示的天数
$showday=date("Y-m-d",(time()+3600*24*$days));         //信息显示的截止时间
```

说明 信息的有效天数与用户交纳的信息费相关，交费不通过本程序完成，因此，信息的有效天数需要管理员手动添加。

27.9.3　付费供求信息发布模块的实现过程

用户通过单击页面导航区的“付费信息”超链接，进入付费信息发布页面，如图 27.29 所示。填写真实有效的付费信息，单击“发布信息”按钮，程序会先验证用户输入的信息，若验证失败，则返回信息发布页面，进行相应提示。若验证成功，则向数据库中插入记录，完成付费信息的发布操作。

在左侧框架 left.php 页中，添加“付费信息”图像域及表单。其代码如下：

（代码位置：光盘\TM\Instance\27\admin\left.php）

```
<form name="form1" method="post" action="release_content.php" target="mainFrame">
```



27.9　付费供求信息发布模块设计

视频讲解：光盘\TM\Video\第 27 章\付费供求信息发布模块设计.exe

27.9.1　付费供求信息发布模块概述

付费供求信息发布是供求信息网站非常重要的功能，也是供求信息网站的盈利点。企业或用户可以根据自身需要对供求信息先进行付费，付费后由管理员在后台将供求信息发布到相应的信息类别中（共包括 11 个信息类别：招聘信息、求职信息、培训信息、公寓信息、家教信息、车辆信息、物品求购、物品出售、求兑出兑、寻求合作、企业广告）。供求信息成功发布后，管理员需要在后台对发布的供求信息进行审核，如果审核通过后，则显示在前台相应的信息类别网页中。付费供求信息发布的流程如图 27.28 所示。

图 27.28　付费供求信息发布流程图

27.9.2　计算供求信息的有效时间

付费供求信息与免费供求信息不同的是，付费供求信息不仅需要收取一定的费用，而且还需要一定的时间限制，例如，网站要求一个月（按 30 天计算）每条信息交 10 元的信息费，如果用户交纳 20 元，那么信息显示的天数就是 60 天。在前台进行显示时，不需要管理员进行手动管理，而是通过程序直接计算出信息显示的截止时间。

信息显示的截止时间=系统当前日期+信息的有效天数（与用户交纳的信息费相关）。

自动计算信息显示的截止时间的具体代码如下：

```
$days=$_POST[days];                                   //通过表单传值获取信息显示的天数
$showday=date("Y-m-d",(time()+3600*24*$days));        //信息显示的截止时间
```

说明　信息的有效天数与用户交纳的信息费相关，交费不通过本程序完成，因此，信息的有效天数需要管理员手动添加。

27.9.3　付费供求信息发布模块的实现过程

用户通过单击页面导航区的“付费信息”超链接，进入付费信息发布页面，如图 27.29 所示。填写真实有效的付费信息，单击“发布信息”按钮，程序会先验证用户输入的信息，若验证失败，则返回信息发布页面，进行相应提示。若验证成功，则向数据库中插入记录，完成付费信息的发布操作。

在左侧框架 left.php 页中，添加“付费信息”图像域及表单。其代码如下：

（代码位置：光盘\TM\Instance\27\admin\left.php）

```
<form name="form1" method="post" action="release_content.php" target="mainFrame">
```

27.9　付费供求信息发布模块设计

视频讲解：光盘\TM\Video\第 27 章\付费供求信息发布模块设计.exe

27.9.1　付费供求信息发布模块概述

付费供求信息发布是供求信息网站非常重要的功能，也是供求信息网站的盈利点。企业或用户可以根据自身需要对供求信息先进行付费，付费后由管理员在后台将供求信息发布到相应的信息类别中（共包括 11 个信息类别：招聘信息、求职信息、培训信息、公寓信息、家教信息、车辆信息、物品求购、物品出售、求兑出兑、寻求合作、企业广告）。供求信息成功发布后，管理员需要在后台对发布的供求信息进行审核，如果审核通过后，则显示在前台相应的信息类别网页中。付费供求信息发布的流程如图 27.28 所示。

管理员

发布的信息

管理页面

审核

N

Y

在前台展示发布的信息

图 27.28　付费供求信息发布流程图

27.9.2　计算供求信息的有效时间

付费供求信息与免费供求信息不同的是，付费供求信息不仅需要收取一定的费用，而且还需要一定的时间限制，例如，网站要求一个月（按 30 天计算）每条信息交 10 元的信息费，如果用户交纳 20 元，那么信息显示的天数就是 60 天。在前台进行显示时，不需要管理员进行手动管理，而是通过程序直接计算出信息显示的截止时间。

信息显示的截止时间=系统当前日期+信息的有效天数（与用户交纳的信息费相关）。

自动计算信息显示的截止时间的具体代码如下：

```
$days=$_POST[days];                                    //通过表单传值获取信息显示的天数
$showday=date("Y-m-d",(time()+3600*24*$days));         //信息显示的截止时间
```

说明 信息的有效天数与用户交纳的信息费相关，交费不通过本程序完成，因此，信息的有效天数需要管理员手动添加。

27.9.3　付费供求信息发布模块的实现过程

用户通过单击页面导航区的“付费信息”超链接，进入付费信息发布页面，如图 27.29 所示。填写真实有效的付费信息，单击“发布信息”按钮，程序会先验证用户输入的信息，若验证失败，则返回信息发布页面，进行相应提示。若验证成功，则向数据库中插入记录，完成付费信息的发布操作。

在左侧框架 left.php 页中，添加“付费信息”图像域及表单。其代码如下：

（代码位置：光盘\TM\Instance\27\admin\left.php）

```
<form name="form1" method="post" action="release_content.php" target="mainFrame">
```



27.9　付费供求信息发布模块设计

视频讲解：光盘\TM\Video\第 27 章\付费供求信息发布模块设计.exe

27.9.1　付费供求信息发布模块概述

付费供求信息发布是供求信息网站非常重要的功能，也是供求信息网站的盈利点。企业或用户可以根据自身需要对供求信息先进行付费，付费后由管理员在后台将供求信息发布到相应的信息类别中（共包括 11 个信息类别：招聘信息、求职信息、培训信息、公寓信息、家教信息、车辆信息、物品求购、物品出售、求兑出兑、寻求合作、企业广告）。供求信息成功发布后，管理员需要在后台对发布的供求信息进行审核，如果审核通过后，则显示在前台相应的信息类别网页中。付费供求信息发布的流程如图 27.28 所示。

管理员

发布的信息

管理页面

审核

N

Y

在前台展示发布的信息

图 27.28　付费供求信息发布流程图

27.9.2　计算供求信息的有效时间

付费供求信息与免费供求信息不同的是，付费供求信息不仅需要收取一定的费用，而且还需要一定的时间限制，例如，网站要求一个月（按 30 天计算）每条信息交 10 元的信息费，如果用户交纳 20 元，那么信息显示的天数就是 60 天。在前台进行显示时，不需要管理员进行手动管理，而是通过程序直接计算出信息显示的截止时间。

信息显示的截止时间=系统当前日期+信息的有效天数（与用户交纳的信息费相关）。

自动计算信息显示的截止时间的具体代码如下：

```
$days=$_POST[days];                                    //通过表单传值获取信息显示的天数
$showday=date("Y-m-d",(time()+3600*24*$days));          //信息显示的截止时间
```

说明

信息的有效天数与用户交纳的信息费相关，交费不通过本程序完成，因此，信息的有效天数需要管理员手动添加。

27.9.3　付费供求信息发布模块的实现过程

用户通过单击页面导航区的“付费信息”超链接，进入付费信息发布页面，如图 27.29 所示。填写真实有效的付费信息，单击“发布信息”按钮，程序会先验证用户输入的信息，若验证失败，则返回信息发布页面，进行相应提示。若验证成功，则向数据库中插入记录，完成付费信息的发布操作。

在左侧框架 left.php 页中，添加“付费信息”图像域及表单。其代码如下：

（代码位置：光盘\TM\Instance\27\admin\left.php）

```
<form name="form1" method="post" action="release_content.php" target="mainFrame">
```



```
    <input name="imageField" type="image" class="input1" src="images/btn_fufei.gif" width="210" height="39" border="0">
</form>
```



图 27.29　付费供求信息发布页面运行结果

单击“付费信息”按钮，将信息页 release_content.php 中的内容显示在框架显示页 mainFrame 中。

付费供求信息发布页面主要用于发布付费的供求信息，该页面中所涉及到的重要表单元素如表 27.7 所示。

表 27.7　付费供求信息页面所涉及的重要表单元素

名　称	元 素 类 型	重 要 属 性	含　义
form1	form	method="post" action="release_ok.php"	表单
type	select	<select name="type"> <option value="招聘信息">-招聘信息-</option> <option value="求职信息" selected>-求职信息-</option> … <option value="寻人/物启示">-寻人/物启示-</option> </select>	信息类型
flag	checkbox	class="input1"　value="1" checked	“是否付费”复选框
title	text	size="50"	信息标题
content	textarea	cols="55" rows="8"	信息内容
linkman	text	size="30"	联系人
tel	text	size="30"	联系电话
days	text		有效天数
imageField	image	src="images/fa.jpg" onClick="return checkformform);"	“发布信息”按钮

在付费信息发布页面选择要发布的信息类型后，填写真实有效的供求信息。为了避免用户添加空信息，在单击“发布信息”按钮时，应用 JavaScript 脚本自定义一个 checkform()函数，验证提交的表单各元素是否为空值，如果为空，则弹出提示信息，并将焦点定位到为空值的表单元素。提交表单信息到数据处理页，应用 insert…into 语句向免费供求信息表中添加供求信息。如果信息添加成功，则弹出成功的信息提示；否则弹出失败的提示信息。其代码如下：

（代码位置：光盘\TM\Instance\27\admin\release_ok.php）

```
<?php
```

```
    <input name="imageField" type="image" class="input1" src="images/btn_fufei.gif" width="210" height="39"
border="0">
</form>
```



图 27.29　付费供求信息发布页面运行结果

单击“付费信息”按钮，将信息页 release_content.php 中的内容显示在框架显示页 mainFrame 中。

付费供求信息发布页面主要用于发布付费的供求信息，该页面中所涉及到的重要表单元素如表 27.7 所示。

表 27.7　付费供求信息页面所涉及的重要表单元素

名　称	元素类型	重要属性	含　义
form1	form	method="post" action="release_ok.php"	表单
type	select	<select name="type"> <option value="招聘信息">-招聘信息-</option> <option value="求职信息" selected>-求职信息-</option> … <option value="寻人/物启示">-寻人/物启示-</option> </select>	信息类型
flag	checkbox	class="input1"　value="1" checked	“是否付费”复选框
title	text	size="50"	信息标题
content	textarea	cols="55" rows="8"	信息内容
linkman	text	size="30"	联系人
tel	text	size="30"	联系电话
days	text		有效天数
imageField	image	src="images/fa.jpg" onClick="return checkformform);"	“发布信息”按钮

在付费信息发布页面选择要发布的信息类型后，填写真实有效的供求信息。为了避免用户添加空信息，在单击“发布信息”按钮时，应用 JavaScript 脚本自定义一个 checkform()函数，验证提交的表单各元素是否为空值，如果为空，则弹出提示信息，并将焦点定位到为空值的表单元素。提交表单信息到数据处理页，应用 insert…into 语句向免费供求信息表中添加供求信息。如果信息添加成功，则弹出成功的信息提示；否则弹出失败的提示信息。其代码如下：

（代码位置：光盘\TM\Instance\27\admin\release_ok.php）

```
<?php
```

```
Include("../conn/conn.php");                                                    //连接数据库文件
$type=$_POST[type];                                                              //获取信息类型
$flag=$_POST[flag];                                                              //获取付款状态
$title=$_POST[title];                                                            //获取信息标题
$content=$_POST[content];                                                        //获取信息内容
$linkman=$_POST[linkman];                                                        //获取联系人
$days=$_POST[days];                                                              //获取发布时间
$tel=$_POST[tel];                                                                //获取联系电话
$sdate=date("Y-m-d");                                                            //当前系统时间
❶    $showday=date("Y-m-d",(time()+3600*24*$days));                             //获取信息的有效时间
$sql=mysql_query("insert into tb_leaguerinfotype,title,content,linkman,tel,sdate,showday,checkstate)
values('$type','$title','$content','$linkman','$tel','$sdate','$showday',$flag)"); //将付费的供求信息添加到数据表中
if($sql){                                                                        //如果添加操作成功，则弹出提示信息
❷    echo "<script>alert('信息发布成功！'); parent.mainFrame.location.href='release_content.php';</script>";
}else{                                                                           //如果添加操作失败，则弹出提示信息
     echo "<script>alert('信息发布失败！');history.back();</script>";
}
?>
```

说明

❶ date("Y-m-d",(time()+3600*24*$days))：信息的有效时间=当前期日期+付费期限。应用 time()函数获取当前日期时间戳，付费期限的时间戳等于 3600 秒×24 小时×指定天数，并通过 date()函数格式化为指定日期格式。

❷ parent.mainFrame.location.href='release_content.php'：刷新父框架页 release_content.php 中的信息。

27.10 付费信息管理模块设计

视频讲解：光盘\TM\Video\第 27 章\付费信息管理模块设计.exe

27.10.1 付费信息管理模块概述

付费信息管理模块主要包括查看图书列表、付费信息审核和付费信息删除 3 个功能。付费信息管理模块的框架如图 27.30 所示。

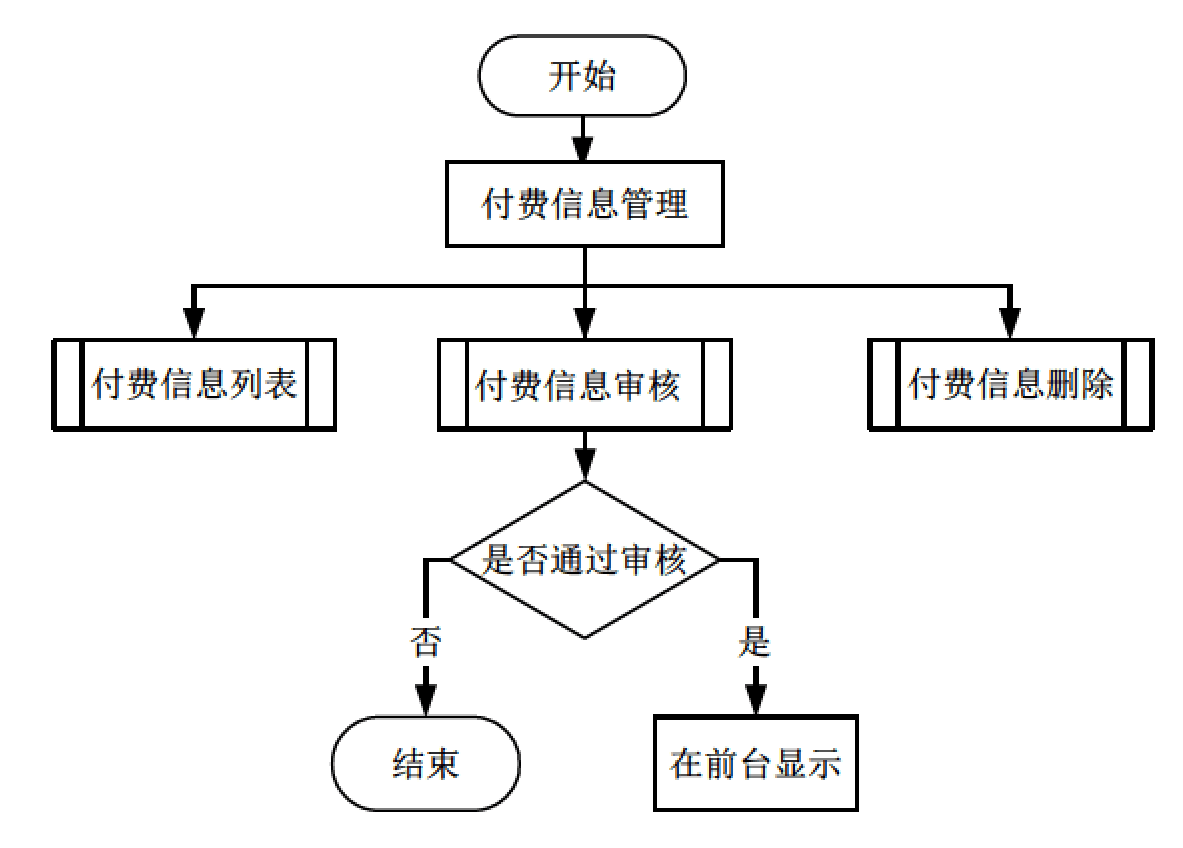


图 27.30 付费信息管理模块的框架图

```
Include("../conn/conn.php");                                    //连接数据库文件
$type=$_POST[type];                                             //获取信息类型
$flag=$_POST[flag];                                             //获取付款状态
$title=$_POST[title];                                           //获取信息标题
$content=$_POST[content];                                       //获取信息内容
$linkman=$_POST[linkman];                                       //获取联系人
$days=$_POST[days];                                             //获取发布时间
$tel=$_POST[tel];                                               //获取联系电话
$sdate=date("Y-m-d");                                           //当前系统时间
❶   $showday=date("Y-m-d",(time()+3600*24*$days));              //获取信息的有效时间
$sql=mysql_query("insert into tb_leaguerinfotype,title,content,linkman,tel,sdate,showday,checkstate)
values('$type','$title','$content','$linkman','$tel','$sdate','$showday',$flag)"); //将付费的供求信息添加到数据表中
if($sql){                                                       //如果添加操作成功，则弹出提示信息
❷   echo "<script>alert('信息发布成功！'); parent.mainFrame.location.href='release_content.php';</script>";
}else{                                                          //如果添加操作失败，则弹出提示信息
    echo "<script>alert('信息发布失败！');history.back();</script>";
}
?>
```

说明

❶ date("Y-m-d",(time()+3600*24*$days))：信息的有效时间=当前期日期+付费期限。应用 time()函数获取当前日期时间戳，付费期限的时间戳等于 3600 秒 × 24 小时 × 指定天数，并通过 date()函数格式化为指定日期格式。

❷ parent.mainFrame.location.href='release_content.php'：刷新父框架页 release_content.php 中的信息。

27.10 付费信息管理模块设计

视频讲解：光盘\TM\Video\第 27 章\付费信息管理模块设计.exe

27.10.1 付费信息管理模块概述

付费信息管理模块主要包括查看图书列表、付费信息审核和付费信息删除 3 个功能。付费信息管理模块的框架如图 27.30 所示。

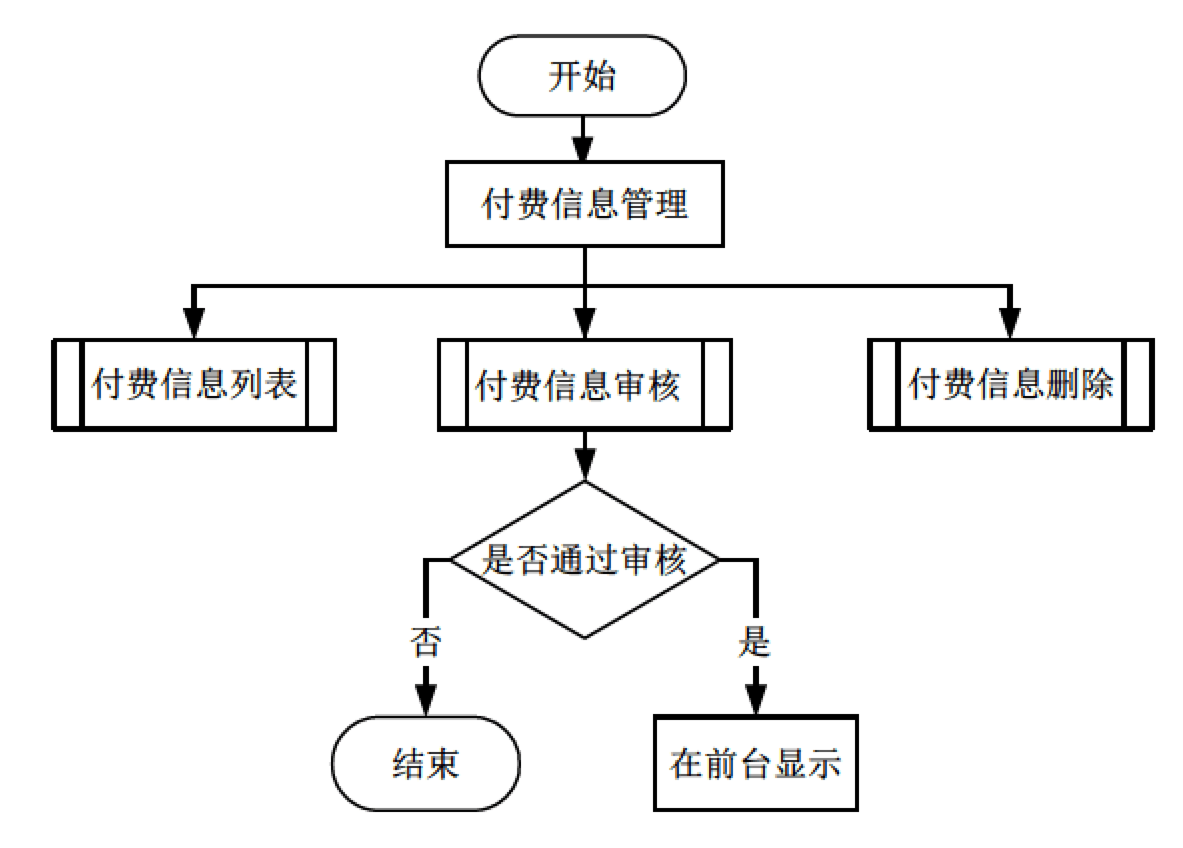


图 27.30 付费信息管理模块的框架图

27.10.2　数据的更新和删除技术

付费信息管理页面在实现信息审核及删除的功能时应用到了 UPDATE 更新语句和 DELETE 删除语句。下面对这两个语句进行详细地讲解。

1．UPDATE 语句

UPDATE 语句用来改变单行上的一列或多列的值，或者改变单个表中选定的一些行上的多个列值。UPDATE 语句的语法如下：

```
UPDATE<table_name | view_name>
SET <column_name>=<expression>
    [….,<last column_name>=<last expression>]
[WHERE<search_condition>]
```

UPDATE 语句的参数说明如表 27.8 所示。

表 27.8　UPDATE 语句的参数说明

参　　数	说　　明
table_name	需要更新的表的名称。如果该表不在当前服务器或数据库中，或不为当前用户所有，这个名称可用链接服务器、数据库和所有者名称来限定
view_name	要更新的视图的名称。通过 view_name 来引用的视图必须是可更新的。用 UPDATE 语句进行的修改，至多只能影响视图的 FROM 子句所引用的基表中的一个
SET	指定要更新的列或变量名称的列表
column_name	含有要更改数据的列的名称。column_name 必须驻留于 UPDATE 子句中所指定的表或视图中。标识列不能进行更新。如果指定了限定的列名称，限定符必须同 UPDATE 子句中的表或视图的名称相匹配
expression	变量、字面值、表达式或加上括号返回单个值的 subSELECT 语句。expression 返回的值将替换 column_name 或 @variable 中的现有值
WHERE	指定条件来限定所更新的行。根据所使用的 WHERE 子句的形式，有两种更新形式： 搜索更新指定搜索条件来限定要删除的行； 定位更新使用 CURRENT OF 子句指定游标。更新操作发生在游标的当前位置
<search_condition>	为要更新行指定需满足的条件。搜索条件也可以是连接所基于的条件。对搜索条件中可以包含的谓词数量没有限制

注意　一定要确保不要忽略 WHERE 子句，除非想要更新表中的所有行。

下面应用 UPDATE 语句将指定职员的工资进行调整，例如，将小璇的基本工资由 2600 元修改为 3000 元。其 SQL 语句如下：

```
update tab_laborage set jbgz=3000 where name='小璇'
```

2．DELETE 语句

DELETE 语句实现删除数据记录。DELETE 语句的语法如下：

```
DELETE FROM <table_name >
[WHERE<search condition>]
```

DELETE 语句的参数说明如表 27.9 所示。

27.10.2　数据的更新和删除技术

付费信息管理页面在实现信息审核及删除的功能时应用到了 UPDATE 更新语句和 DELETE 删除语句。下面对这两个语句进行详细地讲解。

1．UPDATE 语句

UPDATE 语句用来改变单行上的一列或多列的值，或者改变单个表中选定的一些行上的多个列值。UPDATE 语句的语法如下：

```
UPDATE<table_name | view_name>
SET <column_name>=<expression>
    [….,<last column_name>=<last expression>]
[WHERE<search_condition>]
```

UPDATE 语句的参数说明如表 27.8 所示。

表 27.8　UPDATE 语句的参数说明

参　数	说　明
table_name	需要更新的表的名称。如果该表不在当前服务器或数据库中，或不为当前用户所有，这个名称可用链接服务器、数据库和所有者名称来限定
view_name	要更新的视图的名称。通过 view_name 来引用的视图必须是可更新的。用 UPDATE 语句进行的修改，至多只能影响视图的 FROM 子句所引用的基表中的一个
SET	指定要更新的列或变量名称的列表
column_name	含有要更改数据的列的名称。column_name 必须驻留于 UPDATE 子句中所指定的表或视图中。标识列不能进行更新。如果指定了限定的列名称，限定符必须同 UPDATE 子句中的表或视图的名称相匹配
expression	变量、字面值、表达式或加上括号返回单个值的 subSELECT 语句。expression 返回的值将替换 column_name 或 @variable 中的现有值
WHERE	指定条件来限定所更新的行。根据所使用的 WHERE 子句的形式，有两种更新形式： 搜索更新指定搜索条件来限定要删除的行； 定位更新使用 CURRENT OF 子句指定游标。更新操作发生在游标的当前位置
<search_condition>	为要更新行指定需满足的条件。搜索条件也可以是连接所基于的条件。对搜索条件中可以包含的谓词数量没有限制

注意　一定要确保不要忽略 WHERE 子句，除非想要更新表中的所有行。

下面应用 UPDATE 语句将指定职员的工资进行调整，例如，将小璇的基本工资由 2600 元修改为 3000 元。其 SQL 语句如下：

```
update tab_laborage set jbgz=3000 where name='小璇'
```

2．DELETE 语句

DELETE 语句实现删除数据记录。DELETE 语句的语法如下：

```
DELETE FROM <table_name >
[WHERE<search condition>]
```

DELETE 语句的参数说明如表 27.9 所示。

表 27.9　DELETE 语句的参数说明

参　　数	说　　明
FROM	可选的关键字，可用在 DELETE 关键字与目标 table_name、view_name 或 rowset_function_limited 之间
table_name	是要从其中删除行的表的名称
<search_condition>	指定删除行的限定条件。对搜索条件中可以包含的谓词数量没有限制

下面应用 DELETE 语句删除“职员姓名=小璇”的员工基本信息。其 SQL 语句如下：

```
DELETE tab_staffer WHERE ygname='小璇'
```

27.10.3　付费信息显示的实现过程

管理员在后台功能导航区的付费信息显示方式栏中选择相应的信息类别，然后按“已付费”、“未付费”或“全部”中的任意一种状态对付费信息进行管理。例如，在“信息类别”下拉列表框中选择“公寓信息”，在付费状态中选中“全选”单选按钮，单击“检索”按钮提交表单，程序将按指定条件显示出符合条件的所有信息，运行结果如图 27.31 所示。



图 27.31　付费信息显示页面的运行结果

本系统提供了一组单选按钮组成的“付费状态”选项组，付费状态分为已付费、未付费和全部 3 个选项。选中“未付费”单选按钮，则传递的值为“0”；选中“已付费”单选按钮，则传递的值为“1”；选中“全部”单选按钮，则传递的值为“all”。还提供了一个下拉列表框，供用户选择信息类别。将这些单选按钮与下拉列表框都在一个表单中实现，这样，当单击“检索”按钮提交表单后，选择的状态会通过表单进行传递。其表单代码如下：

（代码位置：光盘\TM\Instance\27\admin\left.php）

```
❶  <form name="form3" method="post" action="find_mianfei.php" target="mainFrame">
<tr>
  <td height="65" align="center">
```



```
❷    <fieldset style="height:60;width:210">
          <legend>★ 审核状态</legend>
            <input name="state" type="radio" class="input1" value="1">已审核
            <input name="state" type="radio" class="input1" value="0" checked>未审核
            <input name="state" type="radio" class="input1" value="all"> 全部
    </fieldset>
</td>
</tr>
<tr>
<td height="34" align="center" background="images/info_d.gif">信息类别：
    <select name="type">
      <option value="招聘信息">-招聘信息</option>
      <option value="求职信息" selected>-求职信息</option>
          …
    </select>
     <input type="submit" name="Submit" value="检索">
</td>
  </tr>
</form>
```

说明

❶ target：指定链接的目标窗口，mainFrame 为内容显示区的框架名称。另外，target 有 4 个选项值。

☑ _blank：指定将链接的目标文件加载到未命名的新浏览器窗口中。

☑ _parent：指定将链接的目标文件加载到包含链接的父框架页或窗口中，如果包含链接的框不是嵌套的，则链接的目标文件加载到整个浏览器窗口中。

☑ _self：指定将链接的目标文件加载到链接所在的同一框架或窗口中。

☑ _top：指定将链接的目标文件加载到整个浏览器窗口中，并由此删除所有框架。

❷ <fieldset><legend>…</legend></fieldset>标签：在字符集包含的文本和其他元素外面绘制一个方框。该元素是块元素，必须成对出现。需要注意的是，fieldset 必须用在 form 表单里，一个表单可以有多个<fieldset>…</fieldset>，每对<fieldset></fieldset>为一组，每组的内容描述使用<legend>设置标题名称。

提交表单信息到 find_mianfei.php 页，程序将按管理员选择的指定条件显示出符合条件的所有信息。如果管理员选中“全部”单选按钮，那么系统的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<?php
Include("../conn/conn.php");                          //连接数据库文件
$state=$_POST[payfor];                                //获取管理员选择的付费状态
$type=$_POST[select];                                 //获取管理员选择的信息类别
//如果管理员选择的付费状态为“全部”，则查询管理员指定信息类别下的所有付费信息，并按 id 降序排列
if($state=="all"){
     $sql=mysql_query("select *   from tb_leaguerinfo where type='$type' order by id");
}
else{
//如果管理员选择的付费状态为“已付费”或“未付费”状态，则按管理员指定的条件进行查询，并按 id 降序排列
     $sql=mysql_query("select *   from tb_leaguerinfo   where type='$type' and checkstate=$state order by id");
}
$info=mysql_fetch_array($sql);                        //执行 SQL 语句
?>
```

```
❷    <fieldset style="height:60;width:210">
          <legend>★ 审核状态</legend>
            <input name="state" type="radio" class="input1" value="1">已审核
            <input name="state" type="radio" class="input1" value="0" checked>未审核
            <input name="state" type="radio" class="input1" value="all"> 全部
     </fieldset>
</td>
</tr>
<tr>
<td height="34" align="center" background="images/info_d.gif">信息类别：
     <select name="type">
        <option value="招聘信息">-招聘信息</option>
        <option value="求职信息" selected>-求职信息</option>
            …
     </select>
      <input type="submit" name="Submit" value="检索">
</td>
  </tr>
</form>
```

说明

❶ target：指定链接的目标窗口，mainFrame 为内容显示区的框架名称。另外，target 有 4 个选项值。

☑ _blank：指定将链接的目标文件加载到未命名的新浏览器窗口中。

☑ _parent：指定将链接的目标文件加载到包含链接的父框架页或窗口中，如果包含链接的框不是嵌套的，则链接的目标文件加载到整个浏览器窗口中。

☑ _self：指定将链接的目标文件加载到链接所在的同一框架或窗口中。

☑ _top：指定将链接的目标文件加载到整个浏览器窗口中，并由此删除所有框架。

❷ <fieldset><legend>…</legend></fieldset>标签：在字符集包含的文本和其他元素外面绘制一个方框。该元素是块元素，必须成对出现。需要注意的是，fieldset 必须用在 form 表单里，一个表单可以有多个<fieldset>…</fieldset>，每对<fieldset></fieldset>为一组，每组的内容描述使用<legend>设置标题名称。

提交表单信息到 find_mianfei.php 页，程序将按管理员选择的指定条件显示出符合条件的所有信息。如果管理员选中“全部”单选按钮，那么系统的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<?php
Include("../conn/conn.php");                              //连接数据库文件
$state=$_POST[payfor];                                    //获取管理员选择的付费状态
$type=$_POST[select];                                     //获取管理员选择的信息类别
//如果管理员选择的付费状态为“全部”，则查询管理员指定信息类别下的所有付费信息，并按 id 降序排列
if($state=="all"){
     $sql=mysql_query("select *   from tb_leaguerinfo where type='$type' order by id");
}
else{
//如果管理员选择的付费状态为“已付费”或“未付费”状态，则按管理员指定的条件进行查询，并按 id 降序排列
     $sql=mysql_query("select *   from tb_leaguerinfo   where type='$type' and checkstate=$state order by id");
}
$info=mysql_fetch_array($sql);                            //执行 SQL 语句
?>
```

```
…                                                   //省略供求信息标题的 HTML 代码部分
<?php
if($info){                                          //如果检索到了查询记录
    do{                                             //应用 do…while 循环语句输出付费信息
        if($info[checkstate]==1){                   //如果付费状态的值为 1
            $state1="已付费";                        //则将“已付费”赋给变量$state1
        }else{                                      //如果付费状态的值为 0
            $state1="未付费";                        //则将“未付费”赋给变量$state1
        }
?>
<!-- ---------------------------输出指定符合查询条件的付费信息------------------------------- -->
<tr bgcolor="#FFFFFF">
  <td> <?php echo $info[title];?></td>
  <td width="204"> <?php echo $info[content];?></td>
  <td> <?php echo $info[linkman];?></td>
  <td> <?php echo $info[tel];?></td>
  <td> <?php echo $info[sdate];?></td>
  <td> <?php echo $info[showday];?></td>
  <td align="center" class="style11"><?php echo $state1;?></td>
  <td align="center" bgcolor="#FFFFFF">
<a href="statefu_ok.php?id=<?php echo $info[id];?>&type=<?php echo $type;?>&state=<?php echo
$state;?>" >审核</a>
/<a href="fudel_ok.php?id=<?php echo $info[id];?>&type=<?php echo $type;?>&state=<?php echo $state;?>">
删除</a>
</td>
</tr>
<!-- ------------------------------------------------------------------------------------------ -->
<?php
    }while($info=mysql_fetch_array($sql));          //do…while 循环语句结束
}                                                   //if 条件语句结束

//如果未检索到与管理员指定条件相匹配的记录，则输出相关的提示信息
else{
?>
<tr align="center" bgcolor="#FFFFFF">
    <td colspan="8">对不起，您检索的信息不存在！</td>
  </tr>
<?php
}
?>
```

说明 上面代码中省略了分页显示代码的部分，限于篇幅，参见本书附赠的光盘。

27.10.4 付费信息审核的实现过程

经过审核的信息说明该信息为已付款信息。如果企业或个人用户已登录供求信息但未直接付费，需要后期付款，那么必须进行付款，同时管理员进行审核，经过审核的信息即可在前台进行显示。“审核”超链接的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<a href="statefu_ok.php?id=<?php echo $info[id];?>&type=<?php echo $type;?>&state=<?php echo
$state;?>" >审核</a>
```



```
…                                                                   //省略供求信息标题的 HTML 代码部分
<?php
if($info){                                                          //如果检索到了查询记录
        do{                                                         //应用 do…while 循环语句输出付费信息
                if($info[checkstate]==1){                           //如果付费状态的值为 1
                        $state1="已付费";                           //则将“已付费”赋给变量$state1
                }else{                                              //如果付费状态的值为 0
                        $state1="未付费";                           //则将“未付费”赋给变量$state1
                }
?>
<!-- ---------------------------输出指定符合查询条件的付费信息-------------------------------- -->
<tr bgcolor="#FFFFFF">
   <td> <?php echo $info[title];?></td>
   <td width="204"> <?php echo $info[content];?></td>
   <td> <?php echo $info[linkman];?></td>
   <td> <?php echo $info[tel];?></td>
   <td> <?php echo $info[sdate];?></td>
   <td> <?php echo $info[showday];?></td>
   <td align="center" class="style11"><?php echo $state1;?></td>
   <td align="center" bgcolor="#FFFFFF">
<a   href="statefu_ok.php?id=<?php   echo   $info[id];?>&type=<?php   echo   $type;?>&state=<?php   echo
$state;?>" >审核</a>
/<a href="fudel_ok.php?id=<?php echo $info[id];?>&type=<?php echo $type;?>&state=<?php echo $state;?>">
删除</a>
</td>
</tr>
<!-- ---------------------------------------------------------------------------------------------- -->
<?php
        }while($info=mysql_fetch_array($sql));                      //do…while 循环语句结束
}                                                                   //if 条件语句结束

//如果未检索到与管理员指定条件相匹配的记录，则输出相关的提示信息
else{
?>
<tr align="center" bgcolor="#FFFFFF">
      <td colspan="8">对不起，您检索的信息不存在！</td>
   </tr>
<?php
}
?>
```

说明 上面代码中省略了分页显示代码的部分，限于篇幅，参见本书附赠的光盘。

27.10.4 付费信息审核的实现过程

经过审核的信息说明该信息为已付款信息。如果企业或个人用户已登录供求信息但未直接付费，需要后期付款，那么必须进行付款，同时管理员进行审核，经过审核的信息即可在前台进行显示。“审核”超链接的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<a   href="statefu_ok.php?id=<?php   echo   $info[id];?>&type=<?php   echo   $type;?>&state=<?php   echo
$state;?>" >审核</a>
```



管理员单击对应主题信息后面的“审核”超链接，将信息所对应的 id 值、信息类型及审核状态传递到数据处理页 statefu_ok.php，用 UPDATE 语句将付费状态设置为 1，说明该信息已经付款。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\27\admin\statefu_ok.php）

```
<?php
include("../conn/conn.php");                                    //连接数据库文件
$id=$_GET[id];                                                  //获取信息 id 的值
$type=$_GET[type];                                              //获取信息类型
$state=$_GET[state];                                            //获取信息付费状态
❶    $sql=mysql_query("update tb_leaguerinfo set checkstate=1 where id=$id");
                                                                //更新对应付费信息的状态为已付款
if($sql){                                                       //如果更新操作成功
❷    echo "<script>alert('该信息已经通过审核！
');window.location.href='find_fufei.php?type=$type&state=$state';</script>";    //弹出操作成功提示信息
}
else{                                                               //如果更新操作失败
     echo "<script>alert('该信息审核操作失败！ ');history.back);</script>";      //弹出操作失败信息
}
?>
```

说明

❶ update…set：用来修改指定表中的数据。

❷ type=$type&state=$state：将变量 type 与 state 的值重新传到 find_fufei.php 页，目的是使 find_fufei.php 页中的$type 和$state 变量重新获得信息类型和付费状态值（管理员选择的检索条件），从而返回到付费信息管理页，并更新数据信息。如果数据处理页不对这两个变量进行传值，那么在返回到付费信息管理页 find_fufei.php 页时将会因为检索不到变量的值而出错。

27.10.5　付费信息删除的实现过程

付费信息管理页中“删除”超链接的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<a href="fudel_ok.php?id=<?php echo $info[id];?>&type=<?php echo $type;?>&state=<?php echo $state;?>">
删除</a>
```

管理员单击对应主题信息后面的“审核”超链接，将信息所对应的 id 值、信息类型及审核状态传递到数据处理页 fudel_ok.php，用 DELETE 语句将 id 指定的供求信息删除。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\27\admin\fudel_ok.php）

```
<?php
include("../conn/conn.php");                                 //连接数据库文件
$id=$_GET[id];                                               //获取信息 id 的值
$type=$_GET[type];                                           //获取信息类型
$state=$_GET[state];                                         //获取信息付费状态
$sql=mysql_query("delete from tb_leaguerinfo where id=$id");  //删除对应的供求信息
if($sql){                                                    //如果删除操作成功，则弹出提示信息
     echo "<script>alert('该信息已经删除！ ');window.location.href='find_fufei.php?type=$type&state=$state';
</script>";
}
else{                                                        //如果删除操作失败，则弹出提示信息
```

管理员单击对应主题信息后面的“审核”超链接，将信息所对应的 id 值、信息类型及审核状态传递到数据处理页 statefu_ok.php，用 UPDATE 语句将付费状态设置为 1，说明该信息已经付款。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\27\admin\statefu_ok.php）

```
<?php
include("../conn/conn.php");                                      //连接数据库文件
$id=$_GET[id];                                                    //获取信息 id 的值
$type=$_GET[type];                                                //获取信息类型
$state=$_GET[state];                                              //获取信息付费状态
❶   $sql=mysql_query("update tb_leaguerinfo set checkstate=1 where id=$id");
                                                                  //更新对应付费信息的状态为已付款
if($sql){                                                         //如果更新操作成功
❷   echo "<script>alert('该信息已经通过审核！
');window.location.href='find_fufei.php?type=$type&state=$state';</script>";    //弹出操作成功提示信息
}
else{                                                                 //如果更新操作失败
    echo "<script>alert('该信息审核操作失败！');history.back);</script>";   //弹出操作失败信息
}
?>
```

说明

❶ update…set：用来修改指定表中的数据。

❷ type=$type&state=$state：将变量 type 与 state 的值重新传到 find_fufei.php 页，目的是使 find_fufei.php 页中的$type 和$state 变量重新获得信息类型和付费状态值（管理员选择的检索条件），从而返回到付费信息管理页，并更新数据信息。如果数据处理页不对这两个变量进行传值，那么在返回到付费信息管理页 find_fufei.php 页时将会因为检索不到变量的值而出错。

27.10.5　付费信息删除的实现过程

付费信息管理页中“删除”超链接的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<a href="fudel_ok.php?id=<?php echo $info[id];?>&type=<?php echo $type;?>&state=<?php echo $state;?>">
删除</a>
```

管理员单击对应主题信息后面的“审核”超链接，将信息所对应的 id 值、信息类型及审核状态传递到数据处理页 fudel_ok.php，用 DELETE 语句将 id 指定的供求信息删除。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\27\admin\fudel_ok.php）

```
<?php
include("../conn/conn.php");                                  //连接数据库文件
$id=$_GET[id];                                                //获取信息 id 的值
$type=$_GET[type];                                            //获取信息类型
$state=$_GET[state];                                          //获取信息付费状态
$sql=mysql_query("delete from tb_leaguerinfo where id=$id");  //删除对应的供求信息
if($sql){                                                     //如果删除操作成功，则弹出提示信息
    echo "<script>alert('该信息已经删除！');window.location.href='find_fufei.php?type=$type&state=$state';
</script>";
}
else{                                                         //如果删除操作失败，则弹出提示信息
```

```
        echo "<script>alert('该信息删除操作失败！');history.back();</script>";
}
?>
```

注意 在删除数据后，仍需要将变量$type 和$state 的值重新传递给付费信息管理页 find_fufei.php，目的是返回到付费信息管理页，查看执行删除操作后的状态。

27.10.6 单元测试

在开发完管理员模块后，对该模块进行单元测试。当管理员对付费供求信息进行审核后，审核操作成功了，但却弹出如图 27.32 所示的错误提示。

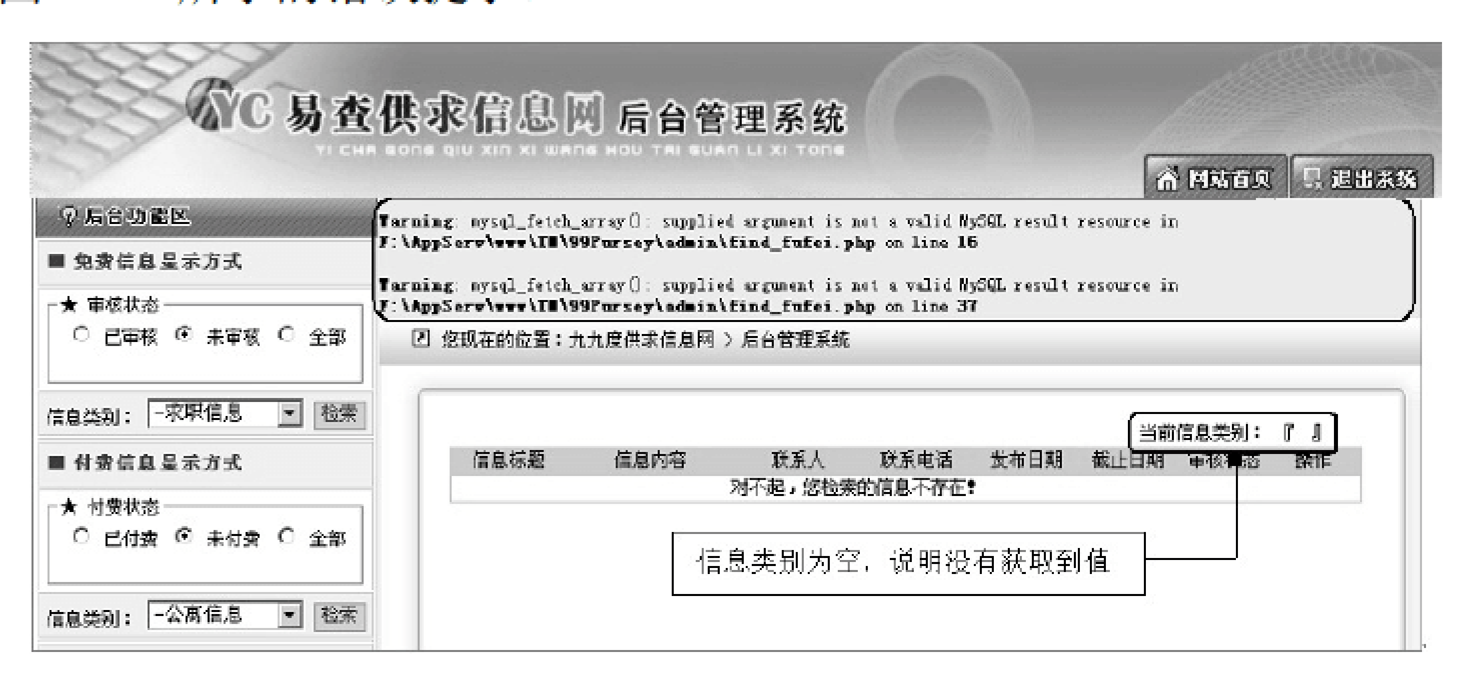


图 27.32 审核付费供求信息的错误提示

在图 27.32 中的错误提示中可以看出，在付费信息管理页的第 16 行和第 37 行出现问题。下面看一下出现问题的这两行代码：

```
if($state=="all"){                          //如果管理员选择的付费状态为“全部”，则执行下面的 SQL 语句
        $sql1=mysql_query("select count(*) as total from tb_leaguerinfo  where type='$type' order by id");
}else{                                      //否则，执行下面的 SQL 语句
        $sql1=mysql_query("select count(*) as total from tb_leaguerinfo  where type='$type' and checkstate=
$state order by id");
}
/*************************第 16 行代码*****************************/
$minfo=mysql_fetch_array($sql1);
/*************************第 17 行代码*****************************/
$total=$minfo[total];
```

从代码中可以看出，SQL 语句的书写并没有错误。根据图 27.32 所示的页面的结果，当前信息类别为空，则说明管理员选择的信息类别没有传过来值，由此可以看出，这是由于在执行审核后页面重新刷新了，因此检索不到管理员选定的查询条件值。

“审核”超链接的源代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<a href="statefu_ok.php?id=<?php echo $info[id];?>">审核</a>
```

审核处理页的源代码如下：

（代码位置：光盘\TM\Instance\27\admin\statefu_ok.php）

```
<?php
$id=$_GET[id];                                              //获取信息 id 的值
        $sql=mysql_query("update tb_leaguerinfo set checkstate=1 where id=$id");
```

```
                                                                        //更新对应付费信息的状态为已付款
if($sql){                                                               //如果更新操作成功，弹出提示信息
    echo "<script>alert('该信息已经通过审核！');window.location.href='find_fufei.php;</script>";
}
```

解决该问题的方法需要在“审核”超链接传值时将管理员选定的“信息类型”和“审核状态”的变量值一同传递到数据处理页，当审核操作完成后，再将“信息类型”和“审核状态”的变量值重新传递给付费信息管理页 find_fufei.php。

“审核”超链接修改后的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<a href="statefu_ok.php?id=<?php echo $info[id];?>&type=<?php echo $type;?>&state=<?php echo $state;?>" >审核</a>
```

技巧　在传递多个变量时，变量之间用“&”符号分隔。

管理员单击对应主题信息后面的“审核”超链接，将信息所对应的 id 值、信息类型及审核状态传递到数据处理页 statefu_ok.php，在执行完更新操作后，需要将“信息类型”和“审核状态”的变量值重新传递到付费信息管理页 find_fufei.php，代码如下。

（代码位置：光盘\TM\Instance\27\admin\statefu_ok.php）

```
<?php
$id=$_GET[id];                                                          //获取信息 id 的值
$type=$_GET[type];                                                      //获取信息类型
$state=$_GET[state];                                                    //获取信息付费状态
    $sql=mysql_query("update tb_leaguerinfo set checkstate=16 where id=$id");
                                                                        //更新对应付费信息的状态为已付款
if($sql){                                                               //如果更新操作成功
    echo "<script>alert('该信息已经通过审核！
');window.location.href='find_fufei.php?type=$type&state=$state';</script>";        //弹出操作成功提示信息
}
```

27.11　小　　结

本章依据软件开发流程介绍了易查供求信息网的开发过程。在开发任何一个项目前，首先要充分做好前期准备，如完善的需求分析、清晰的业务流程、合理的程序结构等，这样，在后期的程序开发中才会得心应手，有备无患。

```
                                                        //更新对应付费信息的状态为已付款
if($sql){                                               //如果更新操作成功，弹出提示信息
    echo "<script>alert('该信息已经通过审核！');window.location.href='find_fufei.php;</script>";
}
```

解决该问题的方法需要在“审核”超链接传值时将管理员选定的“信息类型”和“审核状态”的变量值一同传递到数据处理页，当审核操作完成后，再将“信息类型”和“审核状态”的变量值重新传递给付费信息管理页 find_fufei.php。

“审核”超链接修改后的代码如下：

（代码位置：光盘\TM\Instance\27\admin\find_fufei.php）

```
<a  href="statefu_ok.php?id=<?php  echo  $info[id];?>&type=<?php  echo  $type;?>&state=<?php  echo
$state;?>" >审核</a>
```

技巧 在传递多个变量时，变量之间用“&”符号分隔。

管理员单击对应主题信息后面的“审核”超链接，将信息所对应的 id 值、信息类型及审核状态传递到数据处理页 statefu_ok.php，在执行完更新操作后，需要将“信息类型”和“审核状态”的变量值重新传递到付费信息管理页 find_fufei.php，代码如下。

（代码位置：光盘\TM\Instance\27\admin\statefu_ok.php）

```
<?php
$id=$_GET[id];                                                  //获取信息 id 的值
$type=$_GET[type];                                              //获取信息类型
$state=$_GET[state];                                            //获取信息付费状态
    $sql=mysql_query("update tb_leaguerinfo set checkstate=16 where id=$id");
                                                                //更新对应付费信息的状态为已付款
if($sql){                                                       //如果更新操作成功
    echo "<script>alert('该信息已经通过审核！
');window.location.href='find_fufei.php?type=$type&state=$state';</script>";        //弹出操作成功提示信息
}
```

27.11　小　　结

本章依据软件开发流程介绍了易查供求信息网的开发过程。在开发任何一个项目前，首先要充分做好前期准备，如完善的需求分析、清晰的业务流程、合理的程序结构等，这样，在后期的程序开发中才会得心应手，有备无患。

第28章

图书馆管理系统

（视频讲解：137分钟）

随着网络技术的高速发展和计算机应用的普及，利用计算机对图书馆的日常工作进行管理势在必行。虽然目前很多大型图书馆已经有一整套比较完善的管理系统，但是在一些中小型图书馆中，大部分工作仍需由手工完成，工作效率低，管理员不能及时了解馆内各类图书的借阅情况，读者需要的图书难以在短时间内找到，不便于动态、及时地调整图书结构。为了更好地适应当前读者的借阅需求，解决手工管理中存在的许多问题，越来越多的中小型图书馆正在逐步向计算机信息化管理转变。本章通过开发一个图书馆管理系统，为读者详细讲解项目开发流程。

通过阅读本章内容，你可以：

- 了解图书馆管理系统开发的基本过程
- 掌握系统设计的方法
- 掌握如何分析并设计数据库、数据表
- 掌握多表查询的方法
- 了解主要功能模块的实现方法
- 掌握如何自动计算图书归还日期
- 掌握内联接和外联接语句的使用方法

28.1　图书馆管理系统概述

知源图书馆是师范学校的大型图书馆。随着图书馆规模的不断扩大，图书品种和数量也逐渐增多。但同时，图书馆采用的传统的人工管理方式暴露了一些问题。例如，查找读者借阅的某一本图书的具体摆放位置，需要靠人工记忆在书海中苦苦查找，由于图书存储量大，很难准确定位图书的具体位置，因此每天都要浪费大量宝贵的时间。学校图书馆为提高工作效率，同时摆脱人工管理中出现的种种弊端，现委托某单位开发一个学校图书馆管理系统。

28.2　需 求 分 析

通过计算机对图书进行管理，不仅为图书馆的管理注入了新的生机，而且在运营过程中节省了大量的人力、物力、财力和时间，可以提高图书馆的工作效率，还为图书馆在读者群中树立一个全新的形象，为图书馆日后发展奠定一个良好的基础。通过对一些大型图书馆的实际考察、分析，并结合图书馆的要求以及实际的市场调查，要求本系统具有以下功能。

☑　网站设计页面要求美观大方、个性化，功能全面，操作简单。
☑　要求实现基础信息的管理平台。
☑　要求对所有读者进行管理。
☑　要求实现图书借阅排行，了解当前的畅销书。
☑　商品分类详尽，可按不同类别查看图书信息。
☑　提供快速的图书信息、图书借阅检索功能，保证数据查询的灵活性。
☑　实现图书借阅、图书续借、图书归还的功能。
☑　实现按条件查询，如按用户指定条件查询、按日期时间段查询、按综合条件查询等。
☑　要求记录图书借阅、续借、归还时进行操作的操作员。
☑　实现对图书借阅、续借和归还过程的全程数据信息跟踪。
☑　提供借阅到期提醒功能，使管理者可以及时了解已经到归还日期的图书借阅信息。
☑　提供灵活、方便的权限设置功能，使整个系统的管理分工明确。
☑　具有易维护性和易操作性。

28.3　系 统 设 计

28.3.1　系统目标

根据前面所做的需求分析及用户的需求可以得出，学校图书馆管理系统实施后，应达到以下目标。

☑　网站设计页面要求美观大方、功能全面，操作简单。
☑　网站整体结构和操作流程合理顺畅，实现人性化设计。
☑　规范、完善的基础信息设置。

28.1　图书馆管理系统概述

知源图书馆是师范学校的大型图书馆。随着图书馆规模的不断扩大，图书品种和数量也逐渐增多。但同时，图书馆采用的传统的人工管理方式暴露了一些问题。例如，查找读者借阅的某一本图书的具体摆放位置，需要靠人工记忆在书海中苦苦查找，由于图书存储量大，很难准确定位图书的具体位置，因此每天都要浪费大量宝贵的时间。学校图书馆为提高工作效率，同时摆脱人工管理中出现的种种弊端，现委托某单位开发一个学校图书馆管理系统。

28.2　需 求 分 析

通过计算机对图书进行管理，不仅为图书馆的管理注入了新的生机，而且在运营过程中节省了大量的人力、物力、财力和时间，可以提高图书馆的工作效率，还为图书馆在读者群中树立一个全新的形象，为图书馆日后发展奠定一个良好的基础。通过对一些大型图书馆的实际考察、分析，并结合图书馆的要求以及实际的市场调查，要求本系统具有以下功能。

☑ 网站设计页面要求美观大方、个性化，功能全面，操作简单。
☑ 要求实现基础信息的管理平台。
☑ 要求对所有读者进行管理。
☑ 要求实现图书借阅排行，了解当前的畅销书。
☑ 商品分类详尽，可按不同类别查看图书信息。
☑ 提供快速的图书信息、图书借阅检索功能，保证数据查询的灵活性。
☑ 实现图书借阅、图书续借、图书归还的功能。
☑ 实现按条件查询，如按用户指定条件查询、按日期时间段查询、按综合条件查询等。
☑ 要求记录图书借阅、续借、归还时进行操作的操作员。
☑ 实现对图书借阅、续借和归还过程的全程数据信息跟踪。
☑ 提供借阅到期提醒功能，使管理者可以及时了解已经到归还日期的图书借阅信息。
☑ 提供灵活、方便的权限设置功能，使整个系统的管理分工明确。
☑ 具有易维护性和易操作性。

28.3　系 统 设 计

28.3.1　系统目标

根据前面所做的需求分析及用户的需求可以得出，学校图书馆管理系统实施后，应达到以下目标。

☑ 网站设计页面要求美观大方、功能全面，操作简单。
☑ 网站整体结构和操作流程合理顺畅，实现人性化设计。
☑ 规范、完善的基础信息设置。

- ☑ 对操作员设置不同的操作权限，为管理员提供修改权限。
- ☑ 对所有读者进行集中管理。
- ☑ 对图书信息进行集中管理。
- ☑ 实现图书借阅排行，以便了解当前的畅销书。
- ☑ 提供快速的图书信息、图书借阅检索功能。
- ☑ 实现图书借阅、图书续借、图书归还功能。
- ☑ 实现按条件查询，如按用户指定条件查询、按日期时间段查询、按综合条件查询等。
- ☑ 实现记录图书借阅、续借、归还时进行操作的操作员。
- ☑ 支持图书到期提醒功能。
- ☑ 为操作员提供密码修改功能。
- ☑ 系统运行稳定、安全可靠。

28.3.2　系统功能结构

根据学校图书馆管理系统的特点，可以将其分为系统设置、读者管理、图书档案管理、图书借还和系统查询 5 个部分，其中各个部分及其包括的具体功能模块如图 28.1 所示。

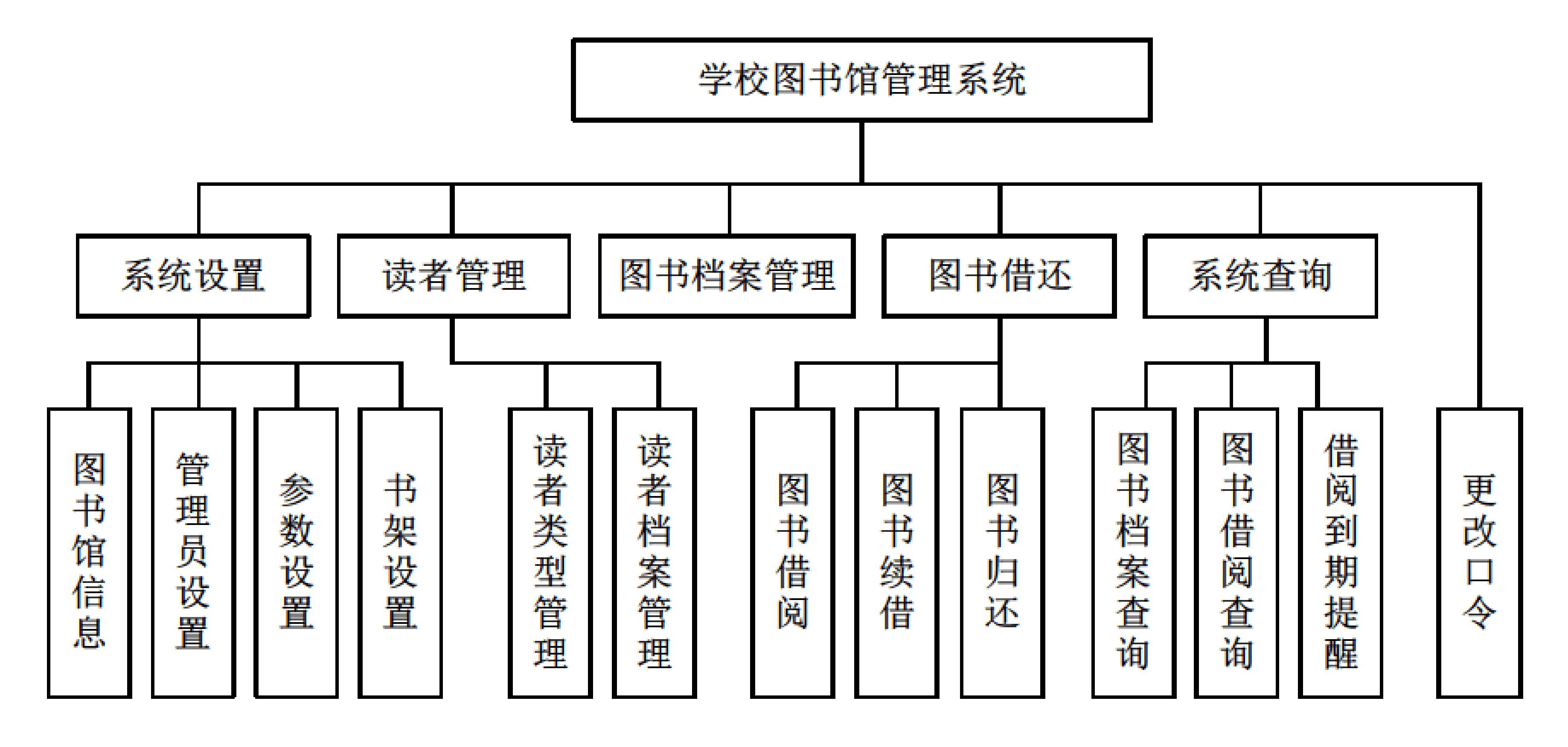


图 28.1　学校图书馆管理系统功能结构图

28.3.3　系统流程图

学校图书馆管理系统的流程如图 28.2 所示。

28.3.4　系统预览

学校图书馆管理系统由多个程序页面组成，下面仅列出几个典型页面，其他页面参见光盘中的源程序。

系统登录页面如图 28.3 所示，该页面用于实现管理员登录。系统首页如图 28.4 所示，该页面用于实现显示系统导航、图书借阅排行和版权信息等功能。



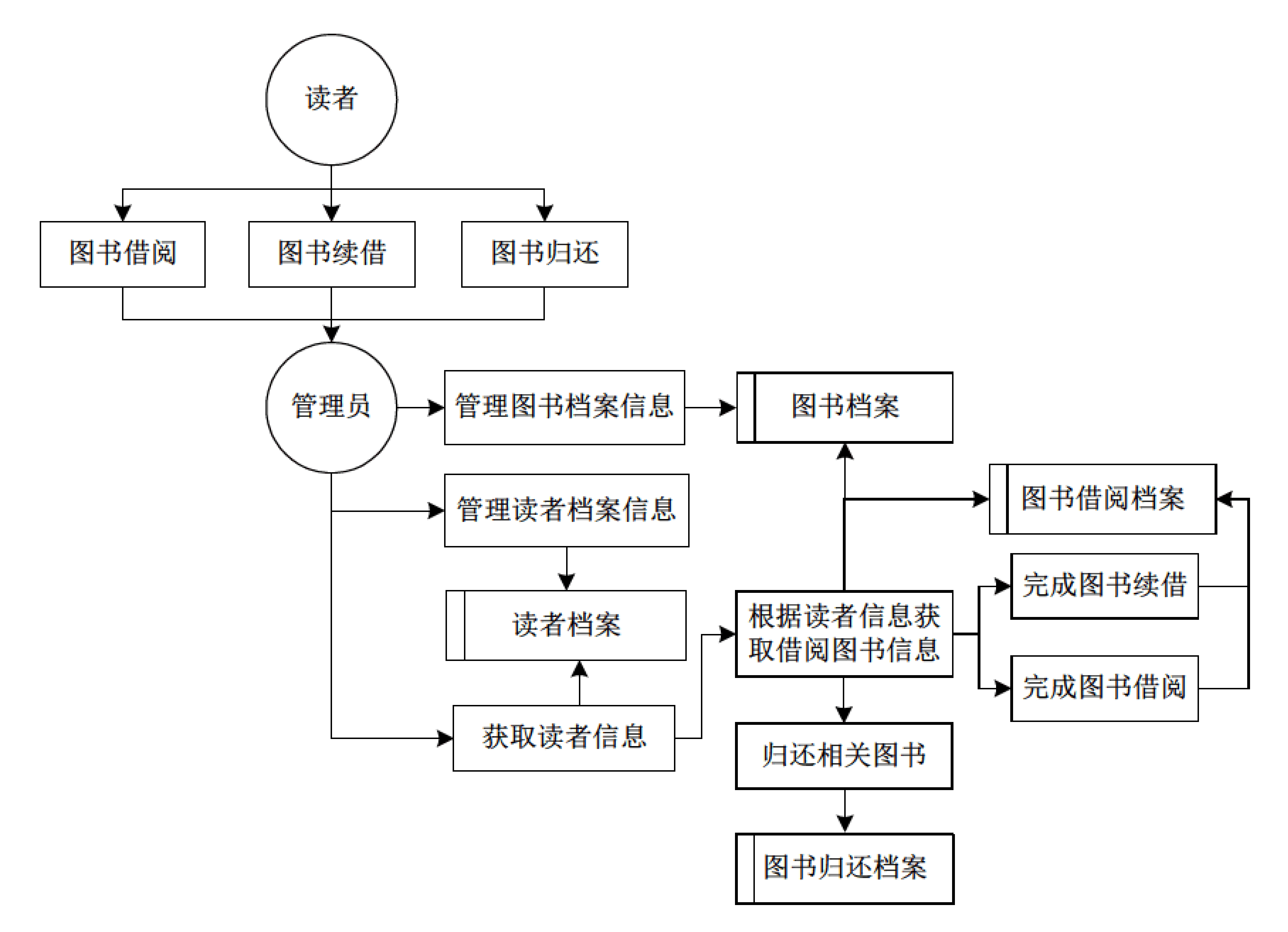


图 28.2　学校图书馆管理系统流程图



图 28.3　系统登录



图 28.4　系统首页

图书借阅页面如图 28.5 所示，该页面用于实现图书借阅功能。图书借阅查询页面如图 28.6 所示，该页面用于实现按照复合条件查询图书借阅信息。



图 28.5　图书借阅



图 28.6　图书借阅查询



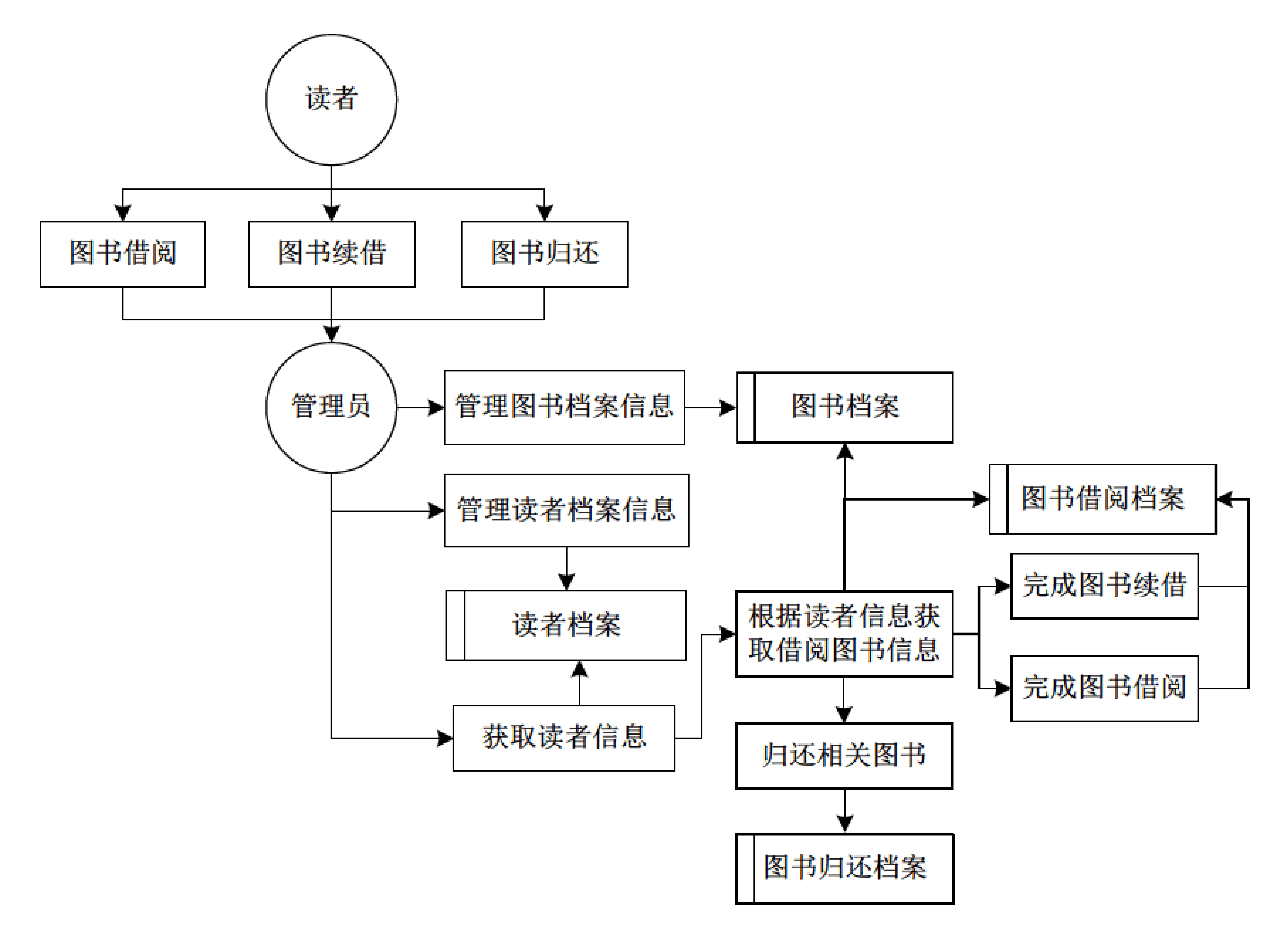


图 28.2　学校图书馆管理系统流程图



图 28.3　系统登录



图 28.4　系统首页

图书借阅页面如图 28.5 所示，该页面用于实现图书借阅功能。图书借阅查询页面如图 28.6 所示，该页面用于实现按照复合条件查询图书借阅信息。



图 28.5　图书借阅



图 28.6　图书借阅查询



28.3.5　文件夹组织结构

视频讲解：光盘\TM\Video\第 28 章\文件夹组织结构.exe

在编写代码前，可以先把系统中可能用到的文件夹创建出来（如创建一个名为 Images 的文件夹，用于保存网站中所使用的图片），这样不但可以方便以后的开发工作，也可以规范网站的整体架构。笔者在开发学校图书馆管理系统时，设计了如图 28.7 所示的文件夹组织结构。在开发时，只需要将所创建的文件保存在相应的文件夹中即可。

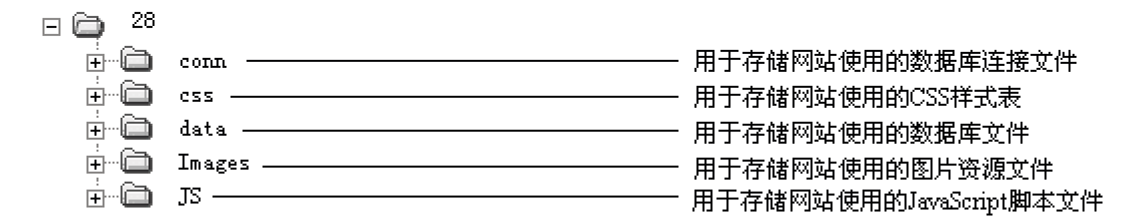


图 28.7　文件夹组织结构

28.4　数据库设计

视频讲解：光盘\TM\Video\第 28 章\数据库设计.exe

学校图书馆管理系统是一个数据库开发的 Web 网站。下面对该系统使用的数据库进行分析和介绍。

28.4.1　数据库分析

由于本系统是为中小型图书馆开发的程序，需要充分考虑成本及使用需求（如跨平台）等问题，而 MySQL 是世界上最为流行的开放源代码的数据库，是完全网络化的跨平台的关系型数据库系统，这正好满足了中小型企业的需求，所以本系统采用 MySQL 数据库。

28.4.2　数据库概念设计

根据对系统所做的需求分析、系统设计，规划出本系统中使用的数据库实体分别为图书档案实体、读者档案实体、借阅档案实体、归还档案实体和管理员实体。下面将介绍几个关键实体的 E-R 图。

1．图书档案实体

图书档案实体包括编号、条形码、书名、类型、作者、译者、出版社、价格、页码、书架、录入时间和操作员等属性。图书档案实体的 E-R 图如图 28.8 所示。

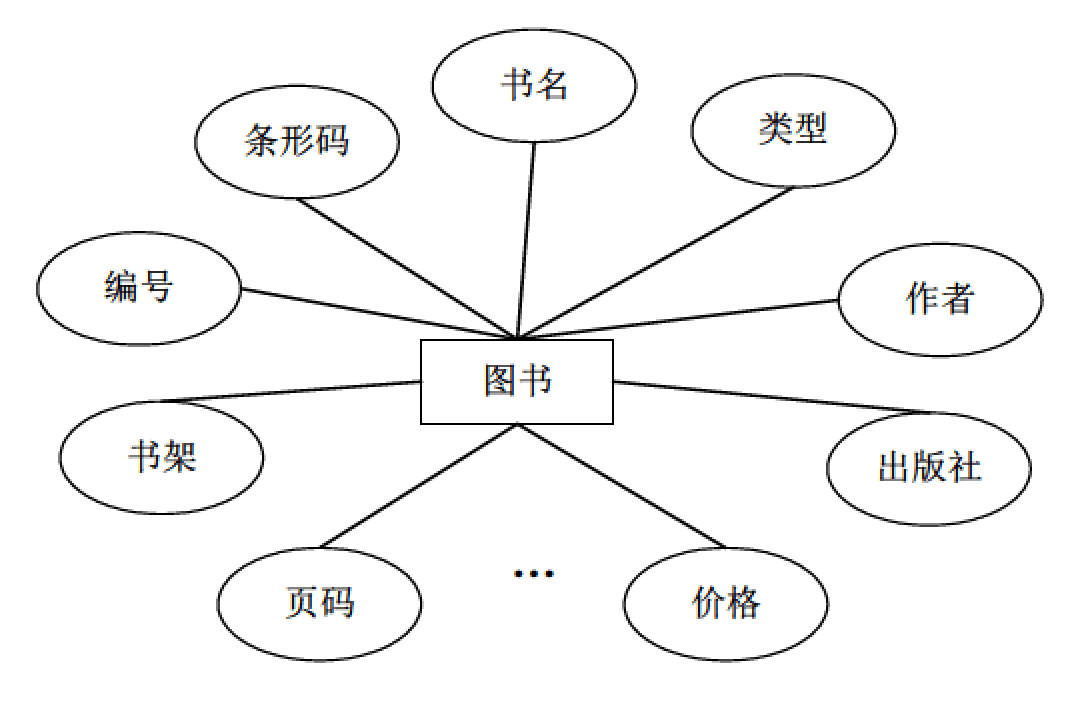


图 28.8　图书档案实体 E-R 图

28.3.5 文件夹组织结构

视频讲解：光盘\TM\Video\第 28 章\文件夹组织结构.exe

在编写代码前，可以先把系统中可能用到的文件夹创建出来（如创建一个名为 Images 的文件夹，用于保存网站中所使用的图片），这样不但可以方便以后的开发工作，也可以规范网站的整体架构。笔者在开发学校图书馆管理系统时，设计了如图 28.7 所示的文件夹组织结构。在开发时，只需要将所创建的文件保存在相应的文件夹中即可。

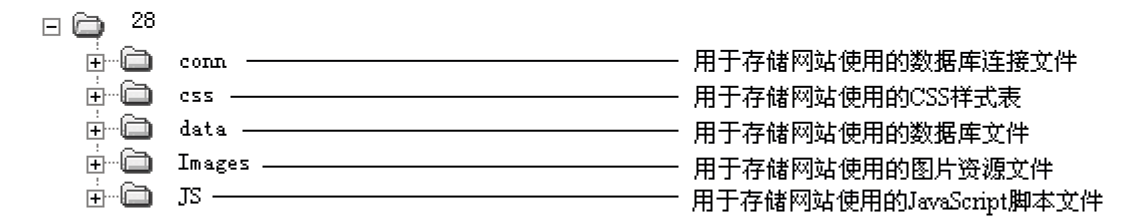


图 28.7 文件夹组织结构

28.4 数据库设计

视频讲解：光盘\TM\Video\第 28 章\数据库设计.exe

学校图书馆管理系统是一个数据库开发的 Web 网站。下面对该系统使用的数据库进行分析和介绍。

28.4.1 数据库分析

由于本系统是为中小型图书馆开发的程序，需要充分考虑成本及使用需求（如跨平台）等问题，而 MySQL 是世界上最为流行的开放源代码的数据库，是完全网络化的跨平台的关系型数据库系统，这正好满足了中小型企业的需求，所以本系统采用 MySQL 数据库。

28.4.2 数据库概念设计

根据对系统所做的需求分析、系统设计，规划出本系统中使用的数据库实体分别为图书档案实体、读者档案实体、借阅档案实体、归还档案实体和管理员实体。下面将介绍几个关键实体的 E-R 图。

1. 图书档案实体

图书档案实体包括编号、条形码、书名、类型、作者、译者、出版社、价格、页码、书架、录入时间和操作员等属性。图书档案实体的 E-R 图如图 28.8 所示。

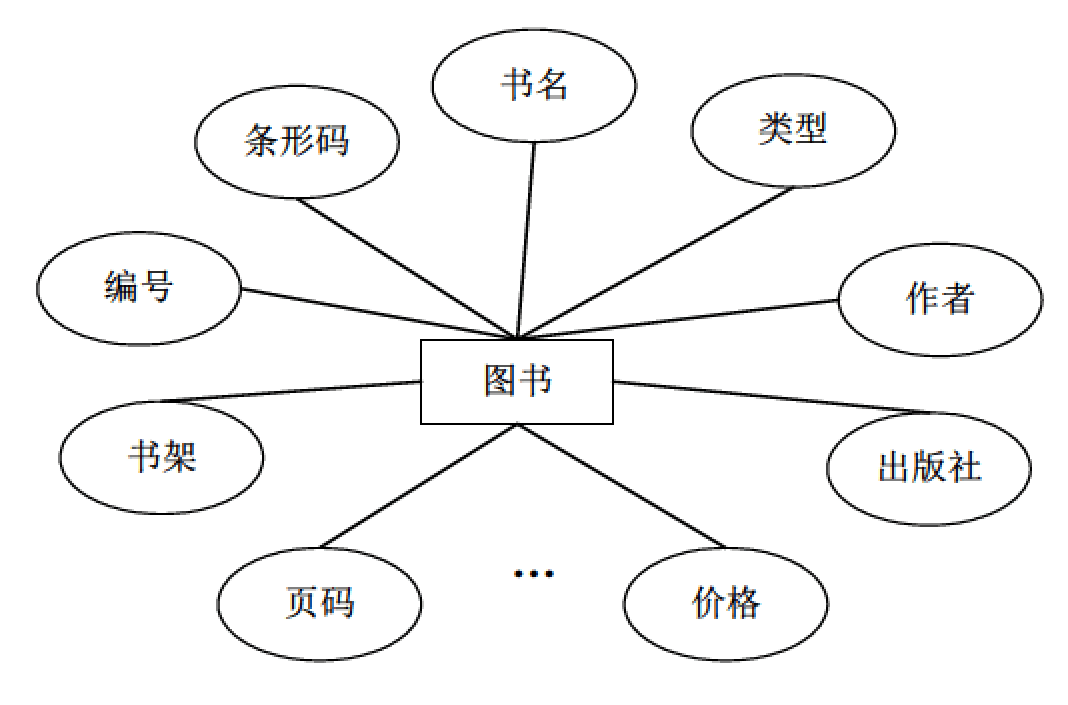


图 28.8 图书档案实体 E-R 图

2．读者档案实体

读者档案实体包括编号、姓名、性别、条形码、职业、出生日期、有效证件、证件号码、电话、电子邮件、登记日期、操作员、类型和备注等属性。读者档案实体的 E-R 图如图 28.9 所示。

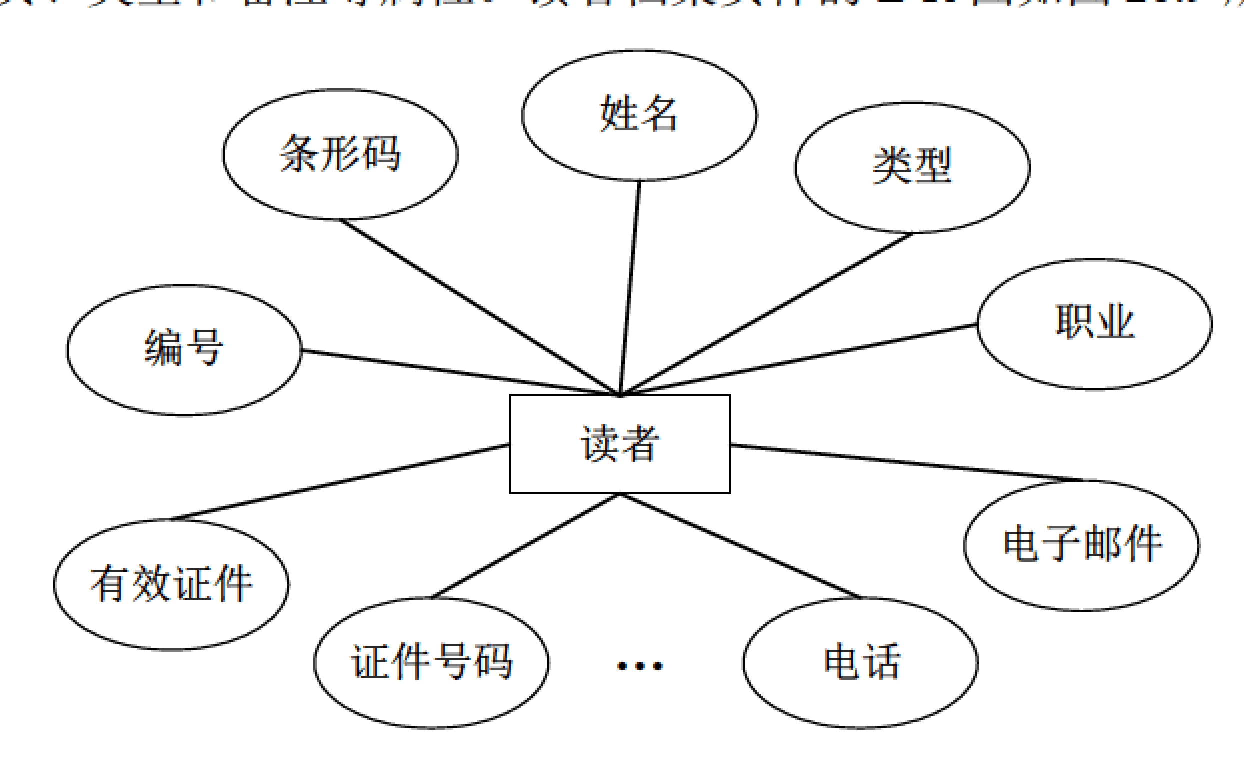


图 28.9　读者档案实体 E-R 图

3．借阅档案实体

借阅档案实体包括编号、读者编号、图书编号、借书时间、应还时间、操作员和是否归还属性。借阅档案实体的 E-R 图如图 28.10 所示。

4．归还档案实体

归还档案实体包括编号、读者编号、图书编号、归还时间和操作员属性。归还档案实体的 E-R 图如图 28.11 所示。

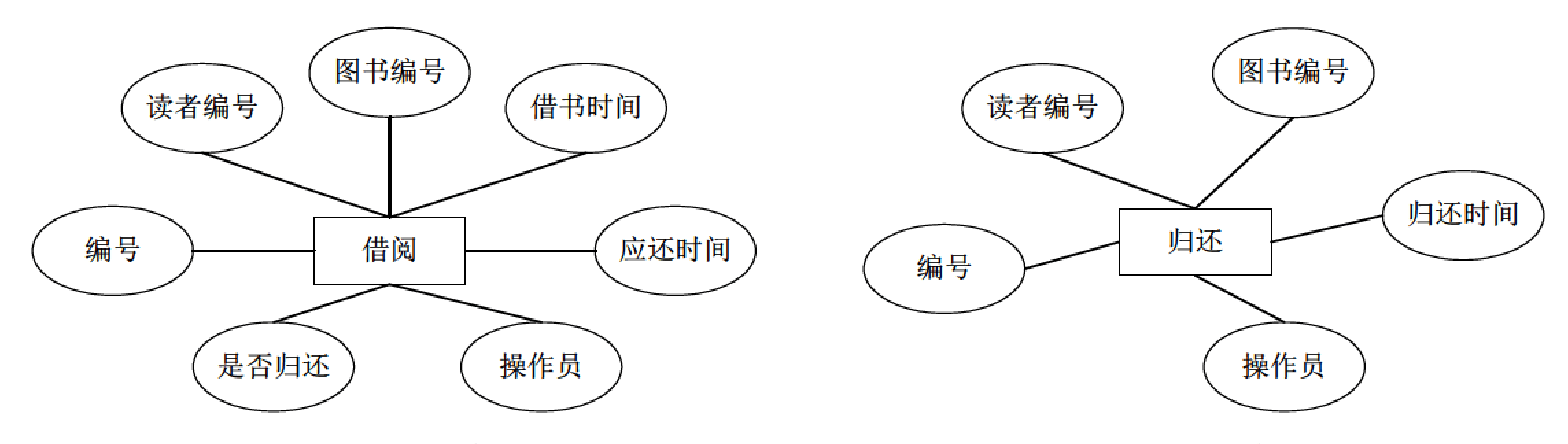


图 28.10　借阅档案实体 E-R 图　　图 28.11　归还档案实体 E-R 图

28.4.3　创建数据库及数据表

结合实际情况及对用户需求的分析，学校图书馆管理系统 db_library 数据库主要包含如下 10 个数据表，如图 28.12 所示。

下面介绍几个主要数据表的结构。

1．tb_bookinfo（图书信息表）

图书信息表主要用于存储图书的基础信息。该数据表的结构如图 28.13 所示。

服务器: localhost ▸ 数据库: db_library

表	类型	整理	说明
tb_bookcase	MyISAM	gb2312_chinese_ci	图书书架信息表
tb_bookinfo	MyISAM	gb2312_chinese_ci	图书信息表
tb_borrow	MyISAM	gb2312_chinese_ci	图书借阅信息表
tb_library	MyISAM	gb2312_chinese_ci	图书馆信息表
tb_manager	MyISAM	gb2312_chinese_ci	管理员信息表
tb_parameter	MyISAM	gb2312_chinese_ci	参数设置信息表
tb_publishing	MyISAM	gb2312_chinese_ci	出版社信息表
tb_purview	MyISAM	gb2312_chinese_ci	权限信息表
tb_reader	MyISAM	gb2312_chinese_ci	读者信息表
tb_readertype	MyISAM	gb2312_chinese_ci	读者类型信息表

图 28.12　学校图书馆管理系统数据表

服务器: localhost ▸ 数据库: db_library ▸ 表 : tb_bookinfo

字段	类型	整理	Null	默认	额外	说明
barcode	varchar(30)	gb2312_chinese_ci	否			图书条形码
bookname	varchar(70)	gb2312_chinese_ci	是	NULL		图书名称
typeid	int(10)		是	NULL		图书类型
author	varchar(30)	gb2312_chinese_ci	是	NULL		图书作者
translator	varchar(30)	gb2312_chinese_ci	是	NULL		图书译者
ISBN	varchar(20)	gb2312_chinese_ci	是	NULL		图书ISBN
price	float(8,2)		是	NULL		图书定价
page	int(10)		是	NULL		图书页码
bookcase	int(10)		是	NULL		图书书架
storage	int(10)		是	NULL		图书库存
inTime	date		是	NULL		入库时间
operator	varchar(30)	gb2312_chinese_ci	是	NULL		操作员
del	tinyint(1)		是	NULL		是否删除
id	int(11)		是	NULL	auto_increment	自动编号id

图 28.13　图书信息表结构

2. tb_borrow（图书借阅信息表）

图书借阅信息表主要用于存储图书的借阅信息。该数据表的结构如图 28.14 所示。

3. tb_reader（读者信息表）

读者信息表主要用于存储读者的基础信息。该数据表的结构如图 28.15 所示。



服务器: localhost ▸ 数据库: db_library ▸ 表 : tb_borrow

字段	类型	整理	Null	默认	额外	说明
id	int(10)		否		auto_increment	自动编号id
readerid	int(10)		是	NULL		读者id编号
bookid	int(10)		是	NULL		图书id编号
borrowTime	date		是	NULL		图书借阅时间
backTime	date		是	NULL		图书归还时间
operator	varchar(30)	gb2312_chinese_ci	是	NULL		操作员
ifback	tinyint(1)		是	0		是否归还

图 28.14　图书借阅信息表结构



服务器: localhost ▸ 数据库: db_library ▸ 表 : tb_reader

字段	类型	整理	Null	默认	额外	说明
id	int(10)		否		auto_increment	自动编号id
name	varchar(20)	gb2312_chinese_ci	是	NULL		读者姓名
sex	varchar(4)	gb2312_chinese_ci	是	NULL		性别
barcode	varchar(30)	gb2312_chinese_ci	是	NULL		读者条形码
vocation	varchar(50)	gb2312_chinese_ci	是	NULL		读者职业
birthday	date		是	NULL		出生日期
paperType	varchar(10)	gb2312_chinese_ci	是	NULL		读者证件
paperNO	varchar(20)	gb2312_chinese_ci	是	NULL		证件号码
tel	varchar(20)	gb2312_chinese_ci	是	NULL		读者电话
email	varchar(100)	gb2312_chinese_ci	是	NULL		E-mail地址
createDate	date		是	NULL		办卡时间
operator	varchar(30)	gb2312_chinese_ci	是	NULL		操作员
remark	mediumtext	gb2312_chinese_ci	是	NULL		备注
typeid	int(11)		是	NULL		读者类型

图 28.15　读者信息表结构



说明 限于篇幅，笔者在此只给出较重要的数据表，其他数据表参见本书附带的光盘。

28.5　首　页　设　计

视频讲解：光盘\TM\Video\第 28 章\首页设计.exe

28.5.1　首页概述

管理员通过系统登录模块的验证后，可以登录到图书馆管理系统的首页。系统首页主要包括导航栏、排行榜和版权信息 3 部分。其中，导航栏中的功能菜单将根据登录管理员的权限进行显示。例如，系统管理员 MR 登录后，将拥有整个系统的全部功能，因为它是超级管理员。

下面看一下本实例中提供的系统首页，该页面在本书光盘中的路径为\TM\28\index.php，如图 28.16 所示。



服务器: localhost ▸ 数据库: db_library

表	类型	整理	说明
tb_bookcase	MyISAM	gb2312_chinese_ci	图书书架信息表
tb_bookinfo	MyISAM	gb2312_chinese_ci	图书信息表
tb_borrow	MyISAM	gb2312_chinese_ci	图书借阅信息表
tb_library	MyISAM	gb2312_chinese_ci	图书馆信息表
tb_manager	MyISAM	gb2312_chinese_ci	管理员信息表
tb_parameter	MyISAM	gb2312_chinese_ci	参数设置信息表
tb_publishing	MyISAM	gb2312_chinese_ci	出版社信息表
tb_purview	MyISAM	gb2312_chinese_ci	权限信息表
tb_reader	MyISAM	gb2312_chinese_ci	读者信息表
tb_readertype	MyISAM	gb2312_chinese_ci	读者类型信息表

图 28.12　学校图书馆管理系统数据表

服务器: localhost ▸ 数据库: db_library ▸ 表 : tb_bookinfo

字段	类型	整理	Null	默认	额外	说明
barcode	varchar(30)	gb2312_chinese_ci	否			图书条形码
bookname	varchar(70)	gb2312_chinese_ci	是	NULL		图书名称
typeid	int(10)		是	NULL		图书类型
author	varchar(30)	gb2312_chinese_ci	是	NULL		图书作者
translator	varchar(30)	gb2312_chinese_ci	是	NULL		图书译者
ISBN	varchar(20)	gb2312_chinese_ci	是	NULL		图书ISBN
price	float(8,2)		是	NULL		图书定价
page	int(10)		是	NULL		图书页码
bookcase	int(10)		是	NULL		图书书架
storage	int(10)		是	NULL		图书库存
inTime	date		是	NULL		入库时间
operator	varchar(30)	gb2312_chinese_ci	是	NULL		操作员
del	tinyint(1)		是	NULL		是否删除
id	int(11)		是	NULL	auto_increment	自动编号id

图 28.13　图书信息表结构

2．tb_borrow（图书借阅信息表）

图书借阅信息表主要用于存储图书的借阅信息。该数据表的结构如图 28.14 所示。

3．tb_reader（读者信息表）

读者信息表主要用于存储读者的基础信息。该数据表的结构如图 28.15 所示。



服务器: localhost ▸ 数据库: db_library ▸ 表 : tb_borrow

字段	类型	整理	Null	默认	额外	说明
id	int(10)		否		auto_increment	自动编号id
readerid	int(10)		是	NULL		读者id编号
bookid	int(10)		是	NULL		图书id编号
borrowTime	date		是	NULL		图书借阅时间
backTime	date		是	NULL		图书归还时间
operator	varchar(30)	gb2312_chinese_ci	是	NULL		操作员
ifback	tinyint(1)		是	0		是否归还

图 28.14　图书借阅信息表结构



服务器: localhost ▸ 数据库: db_library ▸ 表 : tb_reader

字段	类型	整理	Null	默认	额外	说明
id	int(10)		否		auto_increment	自动编号id
name	varchar(20)	gb2312_chinese_ci	是	NULL		读者姓名
sex	varchar(4)	gb2312_chinese_ci	是	NULL		性别
barcode	varchar(30)	gb2312_chinese_ci	是	NULL		读者条形码
vocation	varchar(50)	gb2312_chinese_ci	是	NULL		读者职业
birthday	date		是	NULL		出生日期
paperType	varchar(10)	gb2312_chinese_ci	是	NULL		读者证件
paperNO	varchar(20)	gb2312_chinese_ci	是	NULL		证件号码
tel	varchar(20)	gb2312_chinese_ci	是	NULL		读者电话
email	varchar(100)	gb2312_chinese_ci	是	NULL		E-mail地址
createDate	date		是	NULL		办卡时间
operator	varchar(30)	gb2312_chinese_ci	是	NULL		操作员
remark	mediumtext	gb2312_chinese_ci	是	NULL		备注
typeid	int(11)		是	NULL		读者类型

图 28.15　读者信息表结构



限于篇幅，笔者在此只给出较重要的数据表，其他数据表参见本书附带的光盘。

28.5　首 页 设 计

视频讲解：光盘\TM\Video\第 28 章\首页设计.exe

28.5.1　首页概述

管理员通过系统登录模块的验证后，可以登录到图书馆管理系统的首页。系统首页主要包括导航栏、排行榜和版权信息 3 部分。其中，导航栏中的功能菜单将根据登录管理员的权限进行显示。例如，系统管理员 MR 登录后，将拥有整个系统的全部功能，因为它是超级管理员。

下面看一下本实例中提供的系统首页，该页面在本书光盘中的路径为\TM\28\index.php，如图 28.16 所示。



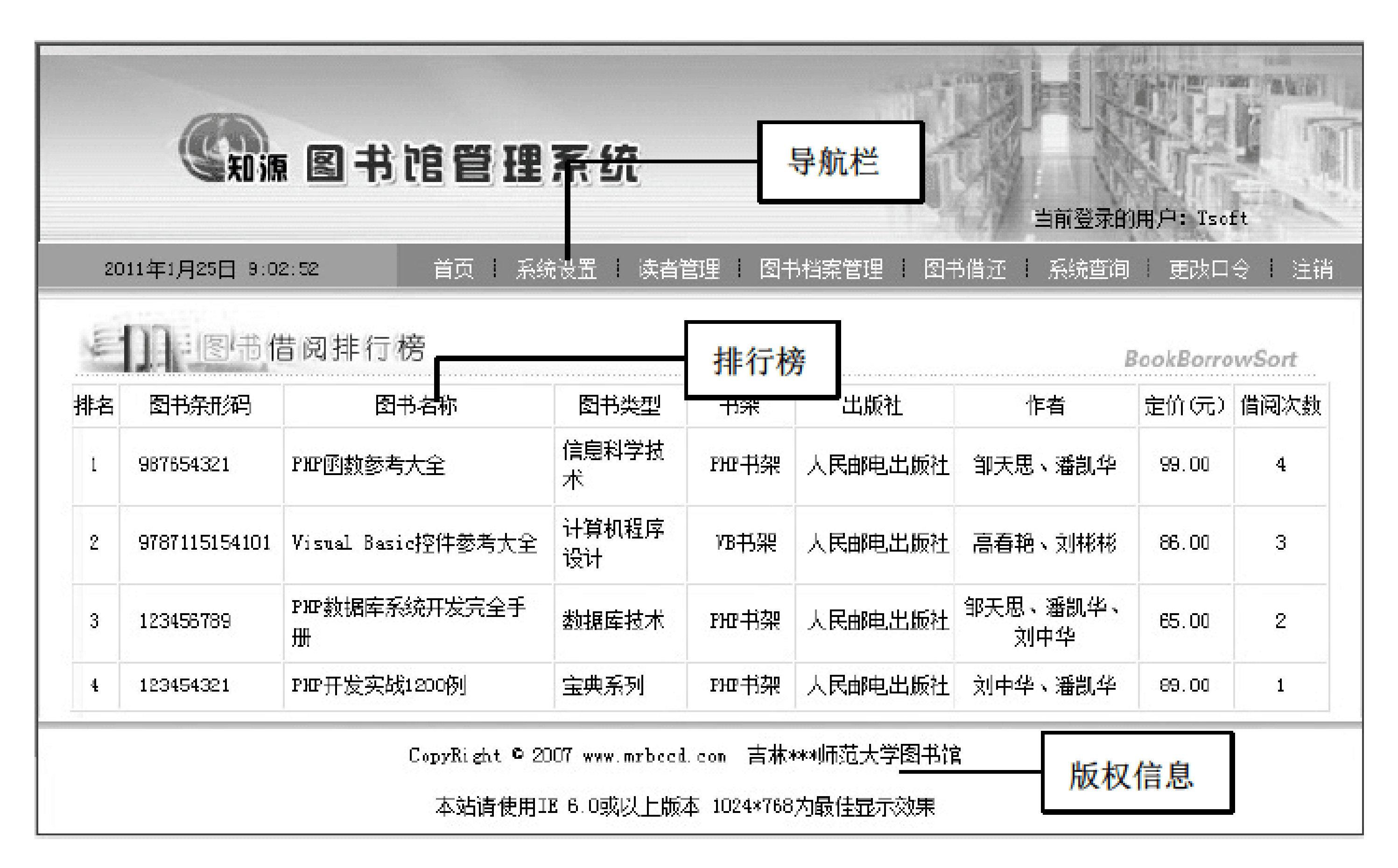


图 28.16　学校图书馆管理系统首页

28.5.2　权限设置技术

学校图书馆管理系统是一个功能全面的大型 Web 网站，出于对网站安全性的考虑，本网站对该系统进行权限的分配，只有管理员级别的超级用户可以对普通用户的权限进行管理和设置。系统首页主要通过判断管理员的权限来显示该用户所操作的功能模块。其关键代码如下：

（代码位置：光盘\TM\Instance\28\navigation.php）

```
<?php
session_start();                                          //初始化 session 变量
include("conn/conn.php");                                 //连接数据库文件
$query=mysql_query("select m.id,m.name,p.id,p.sysset,p.readerset,p.bookset,p.borrowback,p.sysquery from tb_manager as m left join (select * from tb_purview ) as p on m.id=p.id where name='$_SESSION[admin_name]'");
$info=mysql_fetch_array($query);                          //检索用户权限
?>
<!--检索用户所对应的权限，如果权限值为 1，则说明该功能可用，并输出到浏览器，否则不显示-->
<td width="170%" align="right">
<a href="index.php" class="a1">首页</a> ¦
 <?php if($info[sysset]==1){ ?><a onmouseover=showmenu(event,sysmenu) onmouseout=delayhidemenu() style="CURSOR:hand"  class="a1">系统设置</a> ¦ <?php } ?>
<?php if($info[readerset]==1){?><a onmouseover=showmenu(event,readermenu) onmouseout=delayhidemenu() style="CURSOR:hand" class="a1">读者管理</a> ¦ <?php } ?>
<?php if($info[bookset]==1){ ?><a href="book.php" class="a1">图书档案管理</a> ¦ <?php }?>
<?php if($info[borrowback]==1){?><a onmouseover=showmenu(event,borrowmenu) onmouseout=delayhidemenu() style="CURSOR:hand"class="a1" >图书借还</a> ¦ <?php }?>
<?php if($info[sysquery]==1){ ?><a onmouseover=showmenu(event,querymenu) onmouseout=delayhidemenu() style="CURSOR:hand" class="a1">系统查询</a> ¦ <?php } ?>
<a  href="pwd_Modify.php" class="a1">更改口令</a> ¦
<a href="safequit.php" class="a1">注销</a>
</td>
```

说明

在权限信息表 tb_purview 中，权限值为 1，代表具备该模块的操作权限；权限值为 0，代表不具备该模块的操作权限。

在实现系统导航菜单时，引用了 JS 文件 menu.JS，该文件中包含全部实现半透明背景菜单的 JavaScript 代码。

技巧 将页面中所涉及的 JavaScript 代码保存在一个单独的 JS 文件中，然后通过<script></script>引用到需要的页面，可以规范页面代码。在系统导航页面引用 menu.JS 文件的代码如下：

```
<script src="JS/menu.JS"></script>
```

28.5.3 首页的实现过程

系统首页的内容显示区用于显示图书的排行信息，并将排行结果按借阅数量降序排列。该页的关键代码如下：

（代码位置：光盘\TM\Instance\28\index.php）

```
<?php
include("conn/conn.php");                                         //连接数据源文件
$sql=mysql_query("select * from (select bookid,count(bookid) as degree from tb_borrow group by bookid) as
borr join (select b.*,c.name as bookcasename,p.pubname,t.typename from tb_bookinfo b left join tb_bookcase c
on b.bookcase=c.id join tb_publishing p on b.ISBN=p.ISBN join tb_booktype t on b.typeid=t.id where b.del=0) as
book on borr.bookid=book.id order by borr.degree desc limit 10");
$info=mysql_fetch_array($sql);                                    //检索图书借阅信息
$i=1;
do{                                                               //应用 do…while 循环语句显示图书信息
?>
<tr>
	<td height="25" align="center"><?php echo $i;?></td>
	<td style="padding:5px;"> <?php echo $info[barcode];?></td>
	<td style="padding:5px;"><?php echo $info[bookname];?></td>
	<td style="padding:5px;"><?php echo $info[typename];?></td>
	<td align="center"> <?php echo $info[bookcasename];?></td>
	<td align="center"> <?php echo $info[pubname];?></td>
	<td align="center"><?php echo $info[author];?></td>
	<td align="center"><?php echo $info[price];?></td>
	<td align="center"><?php echo $info[degree];?></td>
</tr>
<?php
 $i=$i+1;                                                         //变量自加 1 操作
 }while($info=mysql_fetch_array($sql));                           //do…while 循环语句结束
?>
```

28.6 管理员模块设计

视频讲解：光盘\TM\Video\第 28 章\管理员模块设计.exe

28.6.1 管理员模块概述

管理员模块主要包括管理员登录、查看管理员列表、添加管理员信息、管理员权限设置、删除管理员



在实现系统导航菜单时，引用了 JS 文件 menu.JS，该文件中包含全部实现半透明背景菜单的 JavaScript 代码。

技巧 将页面中所涉及的 JavaScript 代码保存在一个单独的 JS 文件中，然后通过<script></script>引用到需要的页面，可以规范页面代码。在系统导航页面引用 menu.JS 文件的代码如下：

```
<script src="JS/menu.JS"></script>
```

28.5.3 首页的实现过程

系统首页的内容显示区用于显示图书的排行信息，并将排行结果按借阅数量降序排列。该页的关键代码如下：

（代码位置：光盘\TM\Instance\28\index.php）

```
<?php
include("conn/conn.php");                                          //连接数据源文件
$sql=mysql_query("select * from (select bookid,count(bookid) as degree from tb_borrow group by bookid) as
borr join (select b.*,c.name as bookcasename,p.pubname,t.typename from tb_bookinfo b left join tb_bookcase c
on b.bookcase=c.id join tb_publishing p on b.ISBN=p.ISBN join tb_booktype t on b.typeid=t.id where b.del=0) as
book on borr.bookid=book.id order by borr.degree desc limit 10");
$info=mysql_fetch_array($sql);                                     //检索图书借阅信息
$i=1;
do{                                                                //应用 do…while 循环语句显示图书信息
?>
<tr>
    <td height="25" align="center"><?php echo $i;?></td>
    <td style="padding:5px;"> <?php echo $info[barcode];?></td>
    <td style="padding:5px;"><?php echo $info[bookname];?></td>
    <td style="padding:5px;"><?php echo $info[typename];?></td>
    <td align="center"> <?php echo $info[bookcasename];?></td>
    <td align="center"> <?php echo $info[pubname];?></td>
    <td align="center"><?php echo $info[author];?></td>
    <td align="center"><?php echo $info[price];?></td>
    <td align="center"><?php echo $info[degree];?></td>
</tr>
<?php
 $i=$i+1;                                                          //变量自加 1 操作
 }while($info=mysql_fetch_array($sql));                            //do…while 循环语句结束
?>
```

28.6 管理员模块设计

视频讲解：光盘\TM\Video\第 28 章\管理员模块设计.exe

28.6.1 管理员模块概述

管理员模块主要包括管理员登录、查看管理员列表、添加管理员信息、管理员权限设置、删除管理员



和更改口令 6 个功能。管理员模块的框架图如图 28.17 所示。

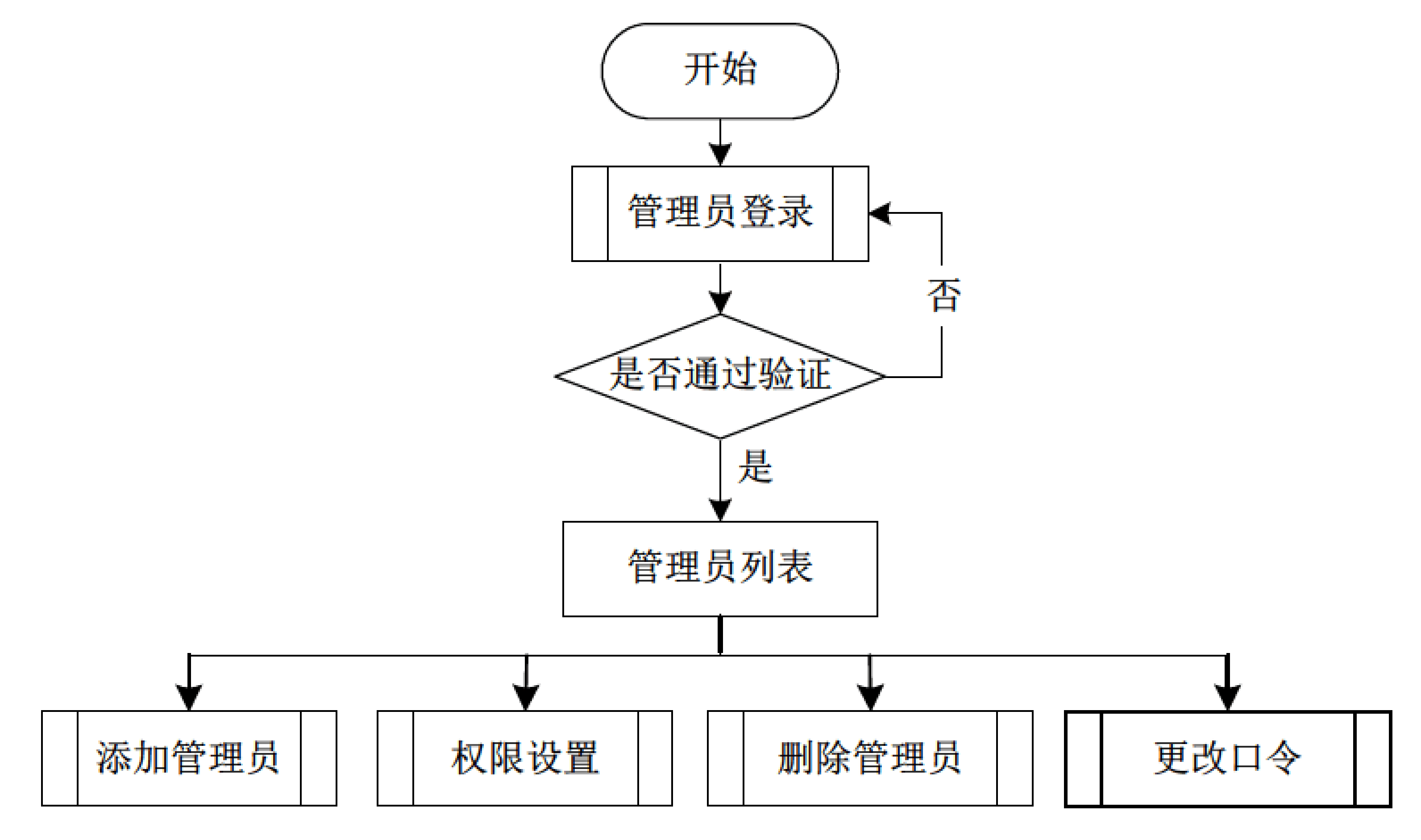


图 28. 17　管理员模块的框架图

28.6.2　控制文件的访问权限

在管理员模块中，涉及到的数据表是 tb_manager（管理员信息表）和 tb_purview（权限表）。其中，管理员信息表中保存的是管理员名称和密码等信息，权限表中保存的是各管理员的权限信息，这两个表通过各自的 id 字段相关联。通过这两个表可以获得完整的管理员信息。

注意 在实现系统登录前，需要在 MySQL 数据库中手动添加一条系统管理员的数据（管理员名为 MR、密码为 mrsoft、拥有所有权限），即在 MySQL 的客户端命令行中应用下面的语句分别向管理员信息表 tb_manager 和权限表 tb_purview 中添加一条数据。

```
#添加管理员信息
insert into tb_manager (name,pwd) values('MR','mrsoft');
#添加权限信息
insert into tb_purview values(1,1,1,1,1,1);
```

从网站安全的角度考虑，仅仅有上面介绍的系统登录页面并不能有效地保证系统的安全，一旦系统首页面的地址被他人获得，就可以通过在地址栏中输入系统的首页面地址而直接进入到系统中。为了便于网站的维护，将验证用户是否登录的代码封装在独立的 PHP 文件中，即 check_login.php 文件。验证用户是否登录的具体代码如下：

（代码位置：光盘\TM\Instance\28\check_login.php）

```
<?php
session_start();
if(!isset($_SESSION['admin_name'])){
     echo "<script>window.location.href='login.php';</script>";
}
?>
```

当系统调用首页时，会判断 session 变量 admin_name 是否存在，如果不存在，将页面重定向到系统登录（login.php）页面。

和更改口令 6 个功能。管理员模块的框架图如图 28.17 所示。

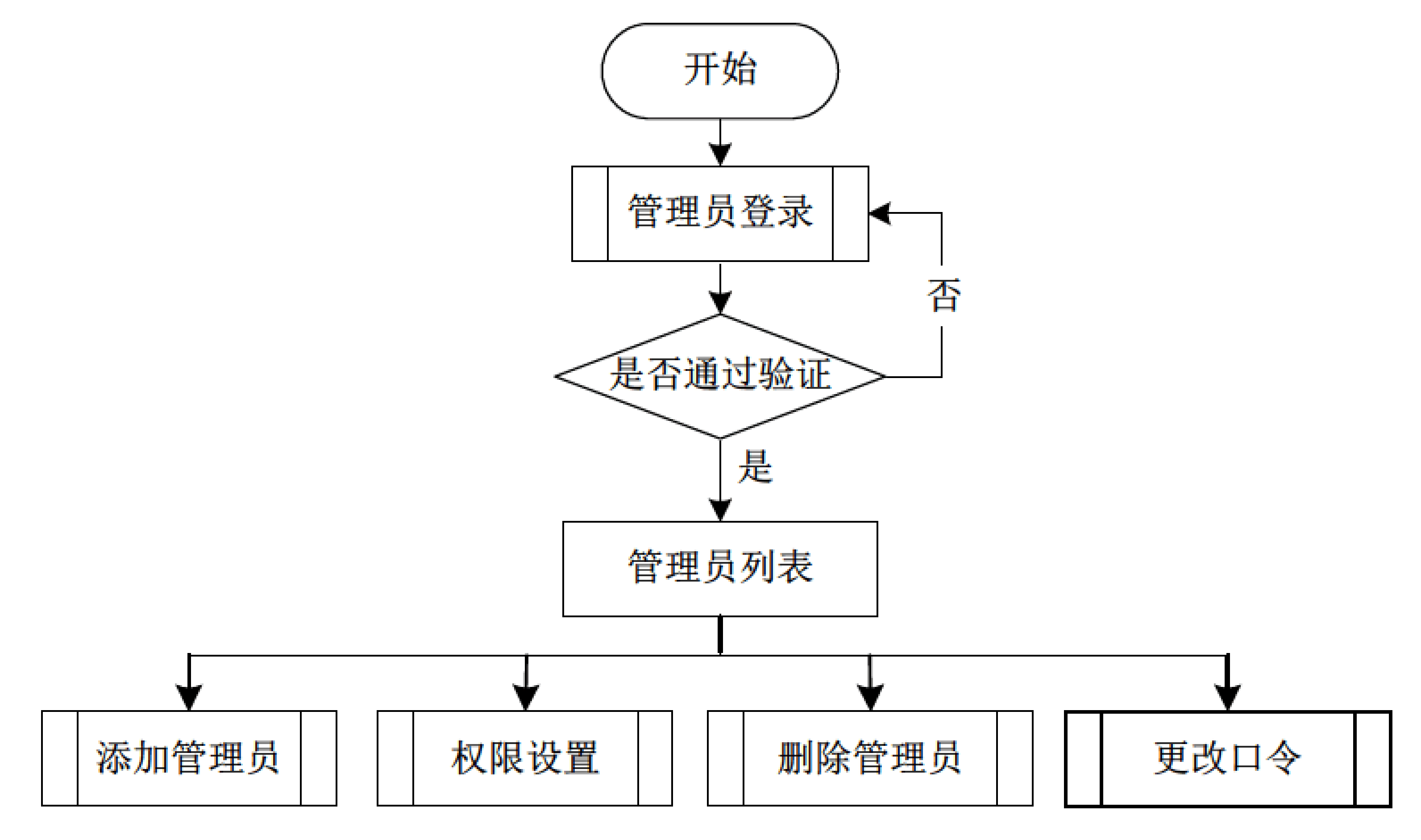


图 28. 17　管理员模块的框架图

28.6.2　控制文件的访问权限

在管理员模块中，涉及到的数据表是 tb_manager（管理员信息表）和 tb_purview（权限表）。其中，管理员信息表中保存的是管理员名称和密码等信息，权限表中保存的是各管理员的权限信息，这两个表通过各自的 id 字段相关联。通过这两个表可以获得完整的管理员信息。

注意 *在实现系统登录前，需要在 MySQL 数据库中手动添加一条系统管理员的数据（管理员名为 MR、密码为 mrsoft、拥有所有权限），即在 MySQL 的客户端命令行中应用下面的语句分别向管理员信息表 tb_manager 和权限表 tb_purview 中添加一条数据。*

```
#添加管理员信息
insert into tb_manager (name,pwd) values('MR','mrsoft');
#添加权限信息
insert into tb_purview values(1,1,1,1,1,1);
```

从网站安全的角度考虑，仅仅有上面介绍的系统登录页面并不能有效地保证系统的安全，一旦系统首页面的地址被他人获得，就可以通过在地址栏中输入系统的首页面地址而直接进入到系统中。为了便于网站的维护，将验证用户是否登录的代码封装在独立的 PHP 文件中，即 check_login.php 文件。验证用户是否登录的具体代码如下：

（代码位置：光盘\TM\Instance\28\check_login.php）

```
<?php
session_start();
if(!isset($_SESSION['admin_name'])){
	echo "<script>window.location.href='login.php';</script>";
}
?>
```

当系统调用首页时，会判断 session 变量 admin_name 是否存在，如果不存在，将页面重定向到系统登录（login.php）页面。

28.6.3　系统登录页面的实现过程

系统登录是进入学校图书馆管理系统的入口，主要用于验证管理员的身份。运行本系统，首先进入的是系统登录页面，在该页面中，系统管理员可以通过输入正确的管理员名称和密码登录到系统首页，当用户没有输入管理员名称或密码时，系统会通过 JavaScript 进行判断，并给予信息提示。系统登录页面的运行结果如图 28.18 所示。



图 28.18　系统登录页面的运行结果

系统登录页面主要用于收集管理员的输入信息及通过自定义的 JavaScript 函数验证输入信息是否为空。该页面所涉及到的表单元素如表 28.1 所示。

表 28.1　系统登录页面所涉及的表单元素

名　称	元 素 类 型	重 要 属 性	含　义
form1	form	method="post" action="chklogin.php"	管理员登录表单
name	text	size="25"	管理员名称
pwd	password	size="25"	管理员密码
submit	submit	value="确定" onclick="return check(form1)"	“确定”按钮
submit3	reset	value="重置"	“重置”按钮
submit2	button	value="关闭" onClick="window.close();"	“关闭”按钮

编写自定义的 JavaScript 函数，用于判断管理员名称和密码是否为空。其代码如下：

（代码位置：光盘\TM\Instance\28\check_login.php）

```
<script language="javascript">
function check(form){                          //自定义一个 JavaScript 函数 check()
      if (form.name.value==""){                //如果管理员名称为空，则弹出提示信息，并重新返回焦点
            alert("请输入管理员名称!");form.name.focus();return false;
      }
      if (form.pwd.value==""){                 //如果管理员密码为空，则弹出提示信息，并重新返回焦点
            alert("请输入密码!");form.pwd.focus();return false;
      }
}
</script>
```

提交表单到数据处理页，页面中为了防止非法用户进入学校图书馆管理系统首页，通过调用 chkinput() 方法判断用户名和密码是否正确。如果为合法用户，则可以登录学校图书馆管理系统的首页；否则，弹出相应的错误提示。其关键代码如下：

（代码位置：光盘\TM\Instance\28\chklogin.php）

```
<?php
session_start();                               //初始化 session 变量
$A_name=$_POST[name];                          //接收表单提交的用户名
$A_pwd=$_POST[pwd];                            //接收表单提交的密码
❶     class chkinput{                          //定义类
    var $name;
    var $pwd;
```

28.6.3　系统登录页面的实现过程

系统登录是进入学校图书馆管理系统的入口，主要用于验证管理员的身份。运行本系统，首先进入的是系统登录页面，在该页面中，系统管理员可以通过输入正确的管理员名称和密码登录到系统首页，当用户没有输入管理员名称或密码时，系统会通过 JavaScript 进行判断，并给予信息提示。系统登录页面的运行结果如图 28.18 所示。



图 28.18　系统登录页面的运行结果

系统登录页面主要用于收集管理员的输入信息及通过自定义的 JavaScript 函数验证输入信息是否为空。该页面所涉及到的表单元素如表 28.1 所示。

表 28.1　系统登录页面所涉及的表单元素

名　　称	元 素 类 型	重 要 属 性	含　　义
form1	form	method="post" action="chklogin.php"	管理员登录表单
name	text	size="25"	管理员名称
pwd	password	size="25"	管理员密码
submit	submit	value="确定" onclick="return check(form1)"	“确定”按钮
submit3	reset	value="重置"	“重置”按钮
submit2	button	value="关闭" onClick="window.close();"	“关闭”按钮

编写自定义的 JavaScript 函数，用于判断管理员名称和密码是否为空。其代码如下：

（代码位置：光盘\TM\Instance\28\check_login.php）

```
<script language="javascript">
function check(form){                              //自定义一个 JavaScript 函数 check()
    if (form.name.value==""){                      //如果管理员名称为空，则弹出提示信息，并重新返回焦点
        alert("请输入管理员名称!");form.name.focus();return false;
    }
    if (form.pwd.value==""){                       //如果管理员密码为空，则弹出提示信息，并重新返回焦点
        alert("请输入密码!");form.pwd.focus();return false;
    }
}
</script>
```

提交表单到数据处理页，页面中为了防止非法用户进入学校图书馆管理系统首页，通过调用 chkinput() 方法判断用户名和密码是否正确。如果为合法用户，则可以登录学校图书馆管理系统的首页；否则，弹出相应的错误提示。其关键代码如下：

（代码位置：光盘\TM\Instance\28\chklogin.php）

```
<?php
session_start();                                   //初始化 session 变量
$A_name=$_POST[name];                              //接收表单提交的用户名
$A_pwd=$_POST[pwd];                                //接收表单提交的密码
❶  class chkinput{                                 //定义类
   var $name;
   var $pwd;
```



```
    function chkinput($x,$y){                              //定义一个方法
      $this->name=$x;                                      //将管理员名称传给类对象$this->name
      $this->pwd=$y;                                       //将管理员密码传给类对象$this->pwd
    }
    function checkinput(){
      include("conn/conn.php");                            //连接数据库文件
      $sql=mysql_query("select * from tb_manager where name='".$this->name."' and pwd='".$this->pwd."'",$conn);
      $info=mysql_fetch_array($sql);                       //检索管理员名称和密码是否正确
      if($info==false){                                    //如果管理员名称或密码不正确，则弹出相关的提示信息
            echo  "<script  language='javascript'>alert('您输入的管理员名称错误，请重新输入！');history.back();
</script>";
            exit;
        }
       else{                                               //如果管理员名称或密码正确，则弹出的相关提示信息
          echo "<script>alert('管理员登录成功!');window.location='index.php';</script>";
            $_SESSION[admin_name]=$info[name];             //将管理员名称存到$_SESSION[admin_name]变量中
            $_SESSION[pwd]=$info[pwd];                     //将管理员密码存到$_SESSION[pwd]变量中
    }
  }
}
❷          $obj=new chkinput(trim($name),trim($pwd));      //创建对象
❸     $obj->checkinput();                                  //调用类
?>
```

说明

❶ class：创建一个 PHP 类时，必须使用 class 关键字进行声明，该关键字后紧跟类的名称，之后用花括号将类体进行封装。

❷ new：PHP 中应用 new 关键字创建对象。该模块创建了一个名为 chkinput 的验证管理员类。

❸ $obj->checkinput();：通过$obj 对象调用 checkinput 类中的属性和方法，即管理员名称和密码。

28.6.4　查看管理员列表页面的实现过程

管理员登录后，选择“系统设置”/“管理员设置”命令，进入到查看管理员列表页面。在该页面中，将以表格的形式显示全部管理员及其权限信息，并提供添加管理员信息、删除管理员信息和设置管理员权限的超链接。查看管理员列表页面的运行结果如图 28.19 所示。



图 28.19　查看管理员列表页面的运行结果

```
    function chkinput($x,$y){                                //定义一个方法
       $this->name=$x;                                       //将管理员名称传给类对象$this->name
       $this->pwd=$y;                                        //将管理员密码传给类对象$this->pwd
     }
    function checkinput(){
       include("conn/conn.php");                             //连接数据库文件
       $sql=mysql_query("select * from tb_manager where name='".$this->name."' and pwd='".$this->pwd."'",$conn);
       $info=mysql_fetch_array($sql);                        //检索管理员名称和密码是否正确
       if($info==false){                                     //如果管理员名称或密码不正确，则弹出相关的提示信息
             echo "<script language='javascript'>alert('您输入的管理员名称错误，请重新输入！');history.back();
</script>";
             exit;
         }
        else{                                                //如果管理员名称或密码正确，则弹出的相关提示信息
           echo "<script>alert('管理员登录成功!');window.location='index.php';</script>";
             $_SESSION[admin_name]=$info[name];              //将管理员名称存到$_SESSION[admin_name]变量中
             $_SESSION[pwd]=$info[pwd];                      //将管理员密码存到$_SESSION[pwd]变量中
     }
  }
}
❷           $obj=new chkinput(trim($name),trim($pwd));       //创建对象
❸     $obj->checkinput();                                    //调用类
?>
```

说明

❶ class：创建一个 PHP 类时，必须使用 class 关键字进行声明，该关键字后紧跟类的名称，之后用花括号将类体进行封装。

❷ new：PHP 中应用 new 关键字创建对象。该模块创建了一个名为 chkinput 的验证管理员类。

❸ $obj->checkinput();：通过$obj 对象调用 checkinput 类中的属性和方法，即管理员名称和密码。

28.6.4　查看管理员列表页面的实现过程

管理员登录后，选择“系统设置”/“管理员设置”命令，进入到查看管理员列表页面。在该页面中，将以表格的形式显示全部管理员及其权限信息，并提供添加管理员信息、删除管理员信息和设置管理员权限的超链接。查看管理员列表页面的运行结果如图 28.19 所示。



图 28.19　查看管理员列表页面的运行结果

首先使用左外联接语句（left join…on）从数据表 tb_manager 和 tb_purview 中查询出符合条件的数据，然后将查询结果应用 do…while 循环语句输出到浏览器。其关键代码如下：

（代码位置：光盘\TM\Instance\28\manager.php）

```
<?php
include("conn/conn.php");                                     //连接数据库文件
❶    $sql=mysql_query("select  m.id,m.name,p.sysset,p.readerset,p.bookset,p.borrowback,p.sysquery  from
tb_manager as m left join (select * from tb_purview) as p on m.id=p.id");
$info=mysql_fetch_array($sql);                                //检索数据信息
do{                                                           //应用 do…while 循环语句输出查询结果
?>
/* ******************************输出符合查询条件的记录****************************** */
<tr>
    <td style="padding:5px;"><?php echo $info[name];?></td>
❷   <td align="center"><input name="checkbox" type="checkbox" class="noborder" value="checkbox"
disabled="disabled" <?php if($info[sysset]==1){echo ("checked");}?>></td>
    <td align="center"><input name="checkbox" type="checkbox" class="noborder" value="checkbox"
 disabled="disabled" <?php if($info[readerset]==1){echo("checked");}?>></td>
    <td align="center"><input name="checkbox" type="checkbox" class="noborder" value="checkbox" disabled
<?php if($info[bookset]==1){echo("checked");}?>></td>
    <td align="center"><input name="checkbox" type="checkbox" class="noborder" value="checkbox" disabled
<?php if($info[borrowback]==1){echo("checked");}?>></td>
    <td align="center"><input name="checkbox" type="checkbox" class="noborder" value="checkbox" disabled
<?php if($info[sysquery]==1){echo("checked");}?>></td>
    <td align="center"><a href="#" onClick="window.open('manager_modify.php?id=<?php echo
$info[id]; ?>','','width=292,height=1175')">权限设置</a></td>
    <td align="center"><a href="manager_del.php?id=<?php echo $info[id];?>">删除</a></td>
  </tr>
/* ******************************************************************************** */
<?php
  }while($info=mysql_fetch_array($sql));                      //do…while 循环语句结束
?>
```

说明

❶ left join…on：应用左外联接将两个表或多个表连接起来，返回部分或全部匹配行，详解参见 13.4.2 节。

❷ <?php if($info[sysset]==1){echo ("checked");}?>：如果系统设置字段的值为 1，则复选框处于选中状态。

28.6.5 添加管理员信息页面的实现过程

管理员登录后，选择“系统设置”/“管理员设置”命令，进入到查看管理员列表页面，在该页面中单击“添加管理员信息”超链接，打开添加管理员信息页面。添加管理员信息页面的运行结果如图 28.20 所示。

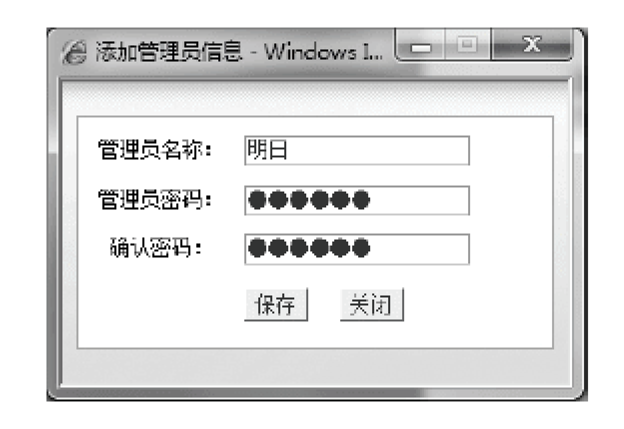


图 28.20 添加管理员页面的运行结果

注意 新添加的管理员信息没有权限，必须通过设置管理员权限为其指定可操作的功能模块。



在查看管理员列表页面，单击“添加管理员信息”超链接的 HTML 代码如下：

（代码位置：光盘\TM\Instance\28\manager.php）

```
<a href="#" onClick="window.open('manager_add.php','','width=292,height=1175')">添加管理员信息</a>
```

添加管理员页面主要用于收集输入的管理员信息及通过自定义的 JavaScript 函数验证输入信息是否合法。该页面所涉及到的表单元素如表 28.2 所示。

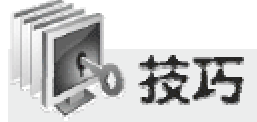
表 28.2　添加管理员页面所涉及的表单元素

名　称	元 素 类 型	重 要 属 性	含　义
form1	form	method="post" action="manager_ok.php"	表单
name	text		管理员名称
pwd	password		管理员密码
pwd1	password		确认密码
submit	submit	value="保存" onClick="check(form1)"	“保存”按钮
Submit2	button	value="关闭" onClick="window.close();"	“关闭”按钮

在添加管理员页面中，输入合法的管理员名称及密码后，单击“保存”按钮，提交表单信息到数据处理页，将添加的管理员信息保存到数据表中。如果添加成功，弹出成功的提示信息；否则，弹出错误提示。其代码如下：

（代码位置：光盘\TM\Instance\28\manager_ok.php）

```
<?php
include("conn/conn.php");                    //连接数据库文件
if($_POST[submit]!=""){                      //如果单击了“保存”按钮，则执行下面的操作
$name=$_POST[name];                          //获取管理员名称
$pwd=$_POST[pwd];                            //获取管理员密码
$sql=mysql_query("insert into tb_manager (name,pwd) values('$name','$pwd')");
if($sql==true){                              //向数据表中添加管理员信息成功，则给出提示信息
echo "<script language=javascript>alert('管理员添加成功！');window.close();window.opener.location.reload();
</script>";
}
else{                                        //向数据表中添加管理员信息失败，则给出提示信息
echo "<script language=javascript>alert('管理员添加失败！');window.close();window.opener.location.reload();
</script>";
}
}
?>
```



在添加管理员处理页中应用“window.opener.location.reload();”语句刷新父窗口中的信息。

28.6.6　设置管理员权限页面的实现过程

在查看管理员列表页面中单击指定管理员后面的“权限设置”超链接，即可进入到权限设置页面，设置该管理员的操作权限。权限设置页面的运行结果如图 28.21 所示。

权限设置页面所涉及到的表单元素如表 28.3 所示。

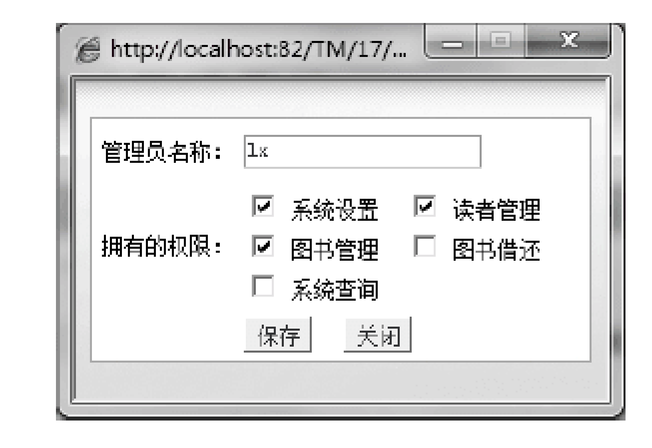


图 28.21　权限设置页面的运行结果

表 28.3　权限设置页面所涉及的表单元素

名　称	元 素 类 型	重 要 属 性	含　义
form1	form	method="post" action="manager_modifyok.php"	表单
id	hidden	value="<?php echo $info[id];?>"	管理员编号
name	text	value="<?php echo $info[name];?>"	管理员名称
sysset	checkbox	<?php if($info[sysset]==1){ echo("checked");}?>	系统设置
readerset	checkbox	<?php if($info[readerset]==1){ echo("checked");}?>	读者管理
bookset	checkbox	<?php if($info[bookset]==1){echo("checked");}?>	图书管理
borrowback	checkbox	<?php if($info[borrowback]==1){echo("checked");}?>	图书借还
sysquery	checkbox	<?php if($info[sysquery]==1){echo("checked");}?>	系统查询
submit	submit	class="btn_grey"	“保存”按钮
Submit2	button	value="关闭" onClick="window.close();"	“关闭”按钮

在查看管理员列表页面中，添加“权限设置”列，并在该列中添加用于打开“权限设置”页面的超链接代码，代码如下：

（代码位置：光盘\TM\Instance\28\manager.php）

```
<a href="#" onClick="window.open('manager_modify.php?id=<?php echo $info[id]; ?>','','width=292, height=1175')">
权限设置</a>
```

从上面的 URL 地址中可以获取设置管理员权限页所涉及到的 id 号，将 id 号提交给处理页 manager_modifyok.php，修改 id 号所对应的管理员信息。其具体代码如下：

（代码位置：光盘\TM\Instance\28\manager_modifyok.php）

```
<?php
include("conn/conn.php");                              //连接数据库文件
if($_POST[submit]!=""){                                //如果提交表单，则执行以下操作
$id=$_POST[id];                                        //获取 id 信息
❶    $sysset=$_POST[sysset]==""?0:1;                   //应用三目运算符求出“系统设置”复选框的值
$readerset=$_POST[readerset]==""?0:1;                  //应用三目运算符求出“读者管理”复选框的值
$bookset=$_POST[bookset]==""?0:1;                      //应用三目运算符求出“图书管理”复选框的值
$borrowback=$_POST[borrowback]==""?0:1;                //应用三目运算符求出“图书借还”复选框的值
$sysquery=$_POST[sysquery]==""?0:1;                    //应用三目运算符求出“系统查询”复选框的值
$query=mysql_query("select * from tb_purview where id=$id");
     $info=mysql_fetch_array($query);                  //检索权限信息表中是否存在该管理员
if($info==false){                                      //如果不存在，向权限表中添加管理员权限信息
❷    mysql_query("insert into tb_purview(id,sysset,readerset,bookset,borrowback,sysquery)
values($id,$sysset,$readerset,$bookset,$borrowback,$sysquery)");
}
else{                                                  //否则，更新管理员的权限信息
     mysql_query("update tb_purview set
 sysset=$sysset,readerset=$readerset,bookset=$bookset,borrowback=$borrowback,sysquery=$sysquery
where id='$id'");
}
echo"<script language=javascript>alert('权限设置修改成功！');window.close();window.opener.location.reload();
</script>";                                            //更新成功，弹出提示信息，并更新父窗口
}
?>
```

表 28.3　权限设置页面所涉及的表单元素

名　称	元素类型	重要属性	含　义
form1	form	method="post" action="manager_modifyok.php"	表单
id	hidden	value="<?php echo $info[id];?>"	管理员编号
name	text	value="<?php echo $info[name];?>"	管理员名称
sysset	checkbox	<?php if($info[sysset]==1){ echo("checked");}?>	系统设置
readerset	checkbox	<?php if($info[readerset]==1){ echo("checked");}?>	读者管理
bookset	checkbox	<?php if($info[bookset]==1){echo("checked");}?>	图书管理
borrowback	checkbox	<?php if($info[borrowback]==1){echo("checked");}?>	图书借还
sysquery	checkbox	<?php if($info[sysquery]==1){echo("checked");}?>	系统查询
submit	submit	class="btn_grey"	“保存”按钮
Submit2	button	value="关闭" onClick="window.close();"	“关闭”按钮

在查看管理员列表页面中，添加“权限设置”列，并在该列中添加用于打开“权限设置”页面的超链接代码，代码如下：

（代码位置：光盘\TM\Instance\28\manager.php）

```
<a href="#" onClick="window.open('manager_modify.php?id=<?php echo $info[id]; ?>','','width=292, height=1175')">
权限设置</a>
```

从上面的 URL 地址中可以获取设置管理员权限页所涉及到的 id 号，将 id 号提交给处理页 manager_modifyok.php，修改 id 号所对应的管理员信息。其具体代码如下：

（代码位置：光盘\TM\Instance\28\manager_modifyok.php）

```
<?php
include("conn/conn.php");                              //连接数据库文件
if($_POST[submit]!=""){                                //如果提交表单，则执行以下操作
$id=$_POST[id];                                        //获取 id 信息
❶    $sysset=$_POST[sysset]==""?0:1;                   //应用三目运算符求出“系统设置”复选框的值
$readerset=$_POST[readerset]==""?0:1;                  //应用三目运算符求出“读者管理”复选框的值
$bookset=$_POST[bookset]==""?0:1;                      //应用三目运算符求出“图书管理”复选框的值
$borrowback=$_POST[borrowback]==""?0:1;                //应用三目运算符求出“图书借还”复选框的值
$sysquery=$_POST[sysquery]==""?0:1;                    //应用三目运算符求出“系统查询”复选框的值
$query=mysql_query("select * from tb_purview where id=$id");
     $info=mysql_fetch_array($query);                  //检索权限信息表中是否存在该管理员
if($info==false){                                      //如果不存在，向权限表中添加管理员权限信息
❷    mysql_query("insert into tb_purview(id,sysset,readerset,bookset,borrowback,sysquery)
values($id,$sysset,$readerset,$bookset,$borrowback,$sysquery)");
}
else{                                                  //否则，更新管理员的权限信息
     mysql_query("update tb_purview set
 sysset=$sysset,readerset=$readerset,bookset=$bookset,borrowback=$borrowback,sysquery=$sysquery
where id='$id'");
}
echo"<script  language=javascript>alert('权限设置修改成功！');window.close();window.opener.location.reload();
</script>";                                            //更新成功，弹出提示信息，并更新父窗口
}
?>
```

说明

❶ $_POST[sysset]==""?0:1;：应用三目运算符求出“系统设置”复选框的值，如果等于空，值为 0，否则值为 1。

❷ insert into：向权限信息表中添加一行数据信息，insert into 语句只适用于对单行数据的插入。

28.6.7　删除管理员的实现过程

在查看管理员列表页面中，单击指定管理员信息后面的“删除”超链接，该管理员及其权限信息将被删除。

在查看管理员列表页面中添加以下用于删除管理员信息的超链接代码。

（代码位置：光盘\TM\Instance\28\manager.php）

```
<a href="manager_del.php?id=<?php echo $info[id];?>">删除</a>
```

从上面的 URL 地址中，可以获取删除管理员所涉及到的 id 号，将 id 号提交给 manager_del.php 处理页，删除 id 号所对应的管理员信息。其具体代码如下：

（代码位置：光盘\TM\Instance\28\manager_del.php）

```
<?php
include("conn/conn.php");                                          //连接数据库文件
$id=$_GET[id];                                                     //获取管理员的 id 号
$sql=mysql_query("delete from tb_manager where id='$id'");         //删除管理员表中 id 号所对应的管理员信息
$query=mysql_query("delete from tb_purview where id='$id'");       //删除权限表中 id 号所对应的管理员权限
if($sql==true and $query==true ){                                  //如果删除操作成功，则弹出提示信息
	echo "<script language=javascript>alert('管理员删除成功！');history.back();</script>";
}
else{                                                              //如果删除操作失败，则弹出提示信息
	echo "<script language=javascript>alert('管理员删除失败！');history.back();</script>";
}
?>
```

28.7　图书档案管理模块设计

视频讲解：光盘\TM\Video\第 28 章\图书档案管理模块设计.exe

28.7.1　图书档案管理模块概述

图书档案管理模块主要包括查看图书列表、添加图书信息、修改图书信息、删除图书信息和查看图书详细信息 5 个功能。图书档案模块的框架如图 28.22 所示。

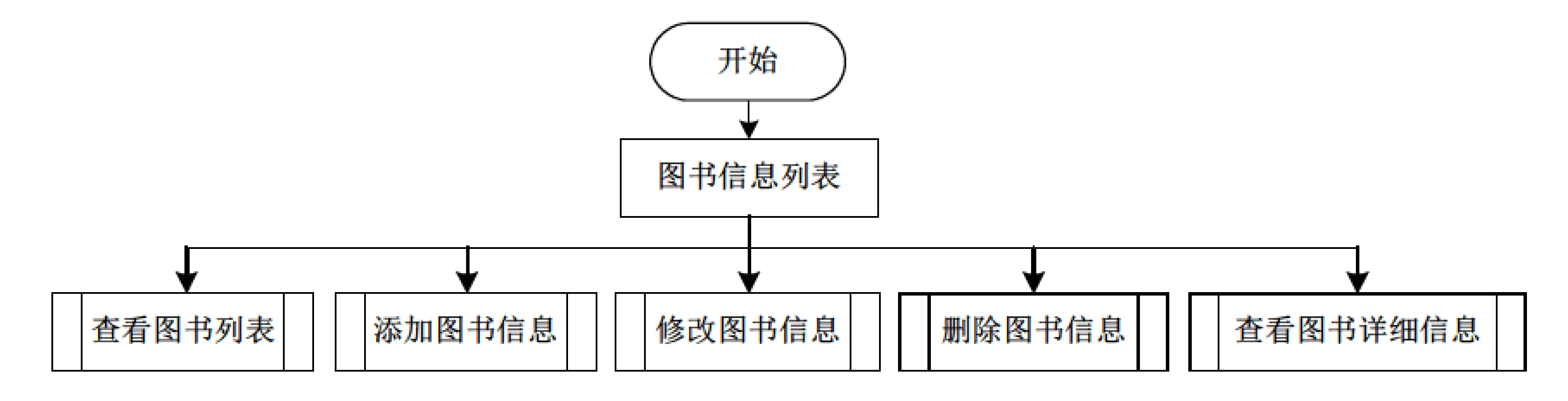


图 28.22　图书档案模块的框架图

28.7.2 图书档案管理中的多表查询技术

在图书档案管理模块中，涉及到的数据表是 tb_bookinfo（图书信息表）、tb_bookcase（书架设置表）、tb_booktype（图书类型表）和 tb_publishing（出版社信息表），这 4 个数据表间通过相应的字段进行关联，如图 28.23 所示，通过以上 4 个表可以获得完整的图书档案信息。

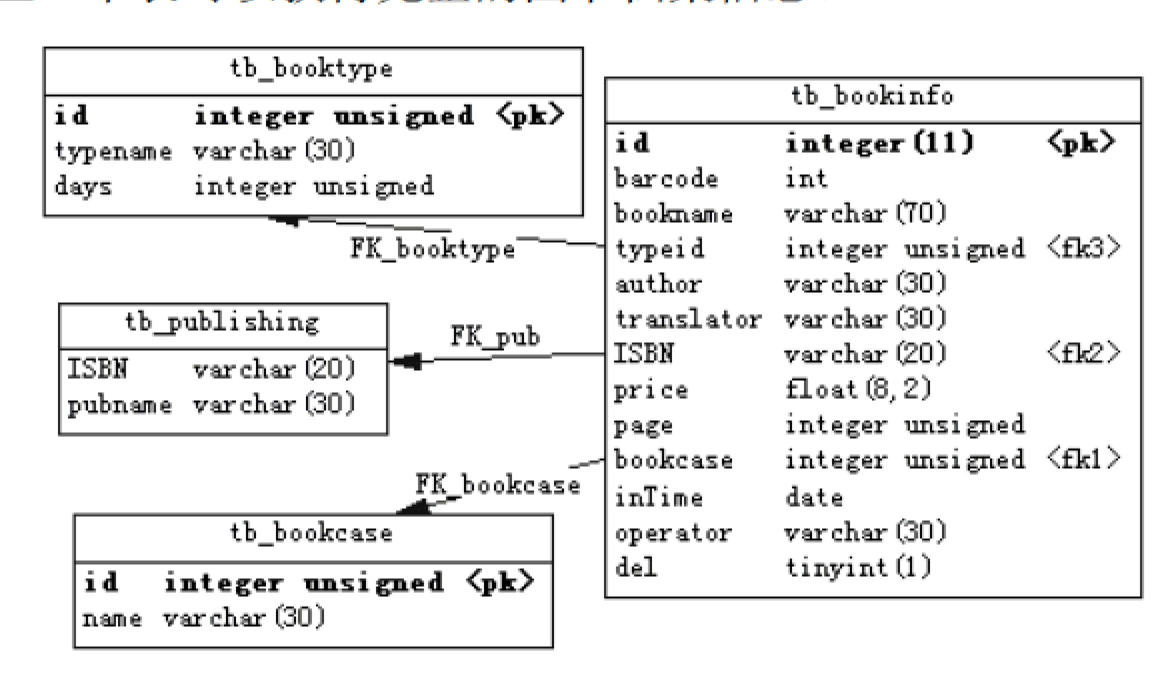


图 28.23 图书档案管理模块各表间关系图

28.7.3 查看图书信息列表页面的实现过程

管理员登录后，选择"图书管理"/"图书档案管理"命令，进入到查看图书列表页面，在该页面中将显示全部图书信息列表，同时提供添加图书信息、删除图书信息、修改图书信息的超链接。查看图书信息列表页面的运行结果如图 28.24 所示。



图 28.24 查看图书信息列表页面的运行结果

打开功能导航文件 navigation.php，设置"图书档案管理"超链接的代码如下：

（代码位置：光盘\TM\Instance\28\navigation.php）

```
<a href="book.php" class="a1">图书档案管理</a>
```

首先应用 join…on 内联接语句将 tb_bookinfo、tb_bookcase、tb_booktype 和 tb_publishing 4 个数据表连接起来，检索指定条件的图书信息，然后应用 do…while 循环语句输出查询结果到浏览器。查看图书信息页面的代码如下：

（代码位置：光盘\TM\Instance\28\book.php）

```
<?php
include("conn/conn.php");                                    //连接数据库文件
```



28.7.2　图书档案管理中的多表查询技术

在图书档案管理模块中，涉及到的数据表是 tb_bookinfo（图书信息表）、tb_bookcase（书架设置表）、tb_booktype（图书类型表）和 tb_publishing（出版社信息表），这 4 个数据表间通过相应的字段进行关联，如图 28.23 所示，通过以上 4 个表可以获得完整的图书档案信息。

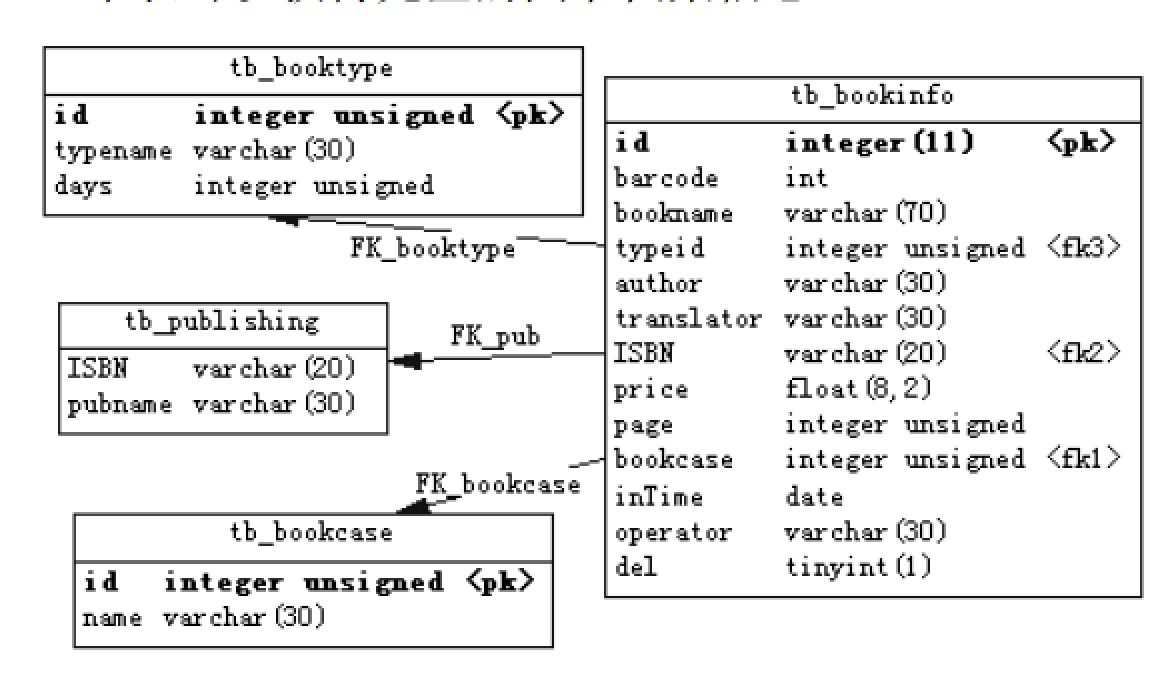


图 28.23　图书档案管理模块各表间关系图

28.7.3　查看图书信息列表页面的实现过程

管理员登录后，选择“图书管理”/“图书档案管理”命令，进入到查看图书列表页面，在该页面中将显示全部图书信息列表，同时提供添加图书信息、删除图书信息、修改图书信息的超链接。查看图书信息列表页面的运行结果如图 28.24 所示。



图 28.24　查看图书信息列表页面的运行结果

打开功能导航文件 navigation.php，设置“图书档案管理”超链接的代码如下：

（代码位置：光盘\TM\Instance\28\navigation.php）

```
<a href="book.php" class="a1">图书档案管理</a>
```

首先应用 join…on 内联接语句将 tb_bookinfo、tb_bookcase、tb_booktype 和 tb_publishing 4 个数据表连接起来，检索指定条件的图书信息，然后应用 do…while 循环语句输出查询结果到浏览器。查看图书信息页面的代码如下：

（代码位置：光盘\TM\Instance\28\book.php）

```
<?php
include("conn/conn.php");                              //连接数据库文件
```



```
$query=mysql_query("select book.barcode,book.id as bookid,book.bookname,bt.typename,pb.pubname,bc.name
from tb_bookinfo book join tb_booktype bt on book.typeid=bt.id join tb_publishing pb on book.ISBN=pb.ISBN
join tb_bookcase bc on book.bookcase=bc.id");
$result=mysql_fetch_array($query);                                //应用内联接检索图书信息
?>
…                                                                 //省略图书信息标题 HTML 标记部分
<?php
 do{                                                              //应用 do…while 循环语句输出查询结果
?>
   <tr>
      <td style="padding:5px;"> <?php echo $result[barcode];?></td>
      <td style="padding:5px;"><a href="book_look.php?id=<?php echo $result[bookid];?>"><?php echo
$result[bookname];?></a></td>
      <td style="padding:5px;"> <?php echo $result[typename];?></td>
      <td style="padding:5px;"> <?php echo $result[pubname];?></td>
      <td style="padding:5px;"> <?php echo $result[name];?></td>
      <td align="center"><a href="book_Modify.php?id=<?php echo $result[bookid];?>">修改</a></td>
      <td align="center"><a href="book_del.php?id=<?php echo $result[bookid];?>">删除</a></td>
   </tr>
<?
   }while($result=mysql_fetch_array($query));                     //do…while 循环语句结束
?>
```

28.7.4　添加图书信息页面的实现过程

管理员登录系统后，在导航栏中单击“图书档案管理”超链接，进入到查看图书列表页面。在该页面中单击“添加图书信息”超链接，进入到添加图书信息页面。添加图书信息页面的运行结果如图 28.25 所示。



图 28.25　添加图书信息页面的运行结果

在查看图书信息列表页面中，设置“添加图书信息”超链接的代码如下：

（代码位置：光盘\TM\Instance\28\book.php）

```
<a href="book_add.php">添加图书信息</a>
```

添加图书信息页面主要用于收集输入的图书信息以及通过自定义的 JavaScript 函数验证输入信息是否合法。该页面中所涉及到的重要表单元素如表 28.4 所示。

表 28.4　添加图书信息页面所涉及的重要表单元素

名　　称	元 素 类 型	重 要 属 性	含　　义
form1	form	method="post" action="book_ok.php"	表单
typeId	select	<?php include("Conn/conn.php"); $sql=mysql_query("select * from tb_booktype"); $info=mysql_fetch_array($sql); do{ ?> <option value="<?php echo $info[id];?>"> <?php echo $info[typename];?></option> <?php }while($info=mysql_fetch_array($sql));?>	图书类型
isbn	select	<?php $sql2=mysql_query("select * from tb_publishing"); 　　$info2=mysql_fetch_array($sql2); 　　do{ ?> <option value="<?php echo $info2[ISBN];?>"> <?php echo $info2[pubname];?></option> <?php }while($info2=mysql_fetch_array($sql2));?>	出版社
bookcaseid	select	<?php $sql3=mysql_query("select * from tb_bookcase"); $info3=mysql_fetch_array($sql3); do{ ?> <option value="<?php echo $info3[id];?>"> <?php echo $info3[name];?></option> <?php }while($info3=mysql_fetch_array($sql3));?>	书架名称
operator	hidden	value="<?php echo $info3[name];?>"	操作员
Submit	submit	onClick="return check(form1)"	"保存"按钮
Submit2	button	onClick="history.back();"	"返回"按钮

由于添加图书信息的方法同添加管理员信息的方法类似，所以此处只给出向图书信息表中插入数据的 SQL 语句，详细代码参见光盘。向图书信息表中插入数据的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\book_ok.php）

```
mysql_query("insert  into
tb_bookinfo(barcode,bookName,typeid,author,translator,ISBN,price,page,bookcase,inTime,operator )values('$b
arcode','$bookName','$typeid','$author','$translator','$isbn','$price','$page','$bookcaseid','$inTime','$operator')");
```

28.7.5　修改图书信息页面的实现过程

管理员登录系统后，在导航栏中单击"图书档案管理"超链接，进入到查看图书列表页面。单击想要



添加图书信息页面主要用于收集输入的图书信息以及通过自定义的 JavaScript 函数验证输入信息是否合法。该页面中所涉及到的重要表单元素如表 28.4 所示。

表 28.4　添加图书信息页面所涉及的重要表单元素

名　　称	元 素 类 型	重 要 属 性	含　　义
form1	form	method="post" action="book_ok.php"	表单
typeId	select	<?php include("Conn/conn.php"); $sql=mysql_query("select * from tb_booktype"); $info=mysql_fetch_array($sql); do{ ?> <option value="<?php echo $info[id];?>"> <?php echo $info[typename];?></option> <?php }while($info=mysql_fetch_array($sql));?>	图书类型
isbn	select	<?php $sql2=mysql_query("select * from tb_publishing"); 　　$info2=mysql_fetch_array($sql2); 　　do{ ?> <option value="<?php echo $info2[ISBN];?>"> <?php echo $info2[pubname];?></option> <?php }while($info2=mysql_fetch_array($sql2));?>	出版社
bookcaseid	select	<?php $sql3=mysql_query("select * from tb_bookcase"); $info3=mysql_fetch_array($sql3); do{ ?> <option value="<?php echo $info3[id];?>"> <?php echo $info3[name];?></option> <?php }while($info3=mysql_fetch_array($sql3));?>	书架名称
operator	hidden	value="<?php echo $info3[name];?>"	操作员
Submit	submit	onClick="return check(form1)"	“保存”按钮
Submit2	button	onClick="history.back();"	“返回”按钮

由于添加图书信息的方法同添加管理员信息的方法类似，所以此处只给出向图书信息表中插入数据的 SQL 语句，详细代码参见光盘。向图书信息表中插入数据的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\book_ok.php）

```
mysql_query("insert  into
tb_bookinfo(barcode,bookName,typeid,author,translator,ISBN,price,page,bookcase,inTime,operator )values('$barcode','$bookName','$typeid','$author','$translator','$isbn','$price','$page','$bookcaseid','$inTime','$operator')");
```

28.7.5　修改图书信息页面的实现过程

管理员登录系统后，在导航栏中单击“图书档案管理”超链接，进入到查看图书列表页面。单击想要



修改的图书信息后面的“修改”超链接，进入到修改图书信息页面。修改图书信息页面的运行结果如图 28.26 所示。

图 28.26　修改图书信息页面的运行结果

在查看图书信息列表页面中，添加“修改”超链接的代码如下：

（代码位置：光盘\TM\Instance\28\book_ok.php）

```
<a href="book_Modify.php?id=<?php echo $result[bookid];?>">修改</a>
```

在修改图书信息页面中修改图书信息后，单击“保存”按钮，提交表单信息到数据处理页 book_Modify_ok.php，应用 UPDATE 语句将修改的图书信息保存到数据表 tb_bookinfo 中，修改成功后，则弹出“图书信息修改成功！”提示信息，并将页面重定向到修改图书信息页。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\28\book_Modify_ok.php）

```
<?php
session_start();                                    //初始化 session 变量
include("conn/conn.php");                           //连接数据库文件
$bid=$_POST[bid];                                   //获取图书 id 号
$operator=$_SESSION[admin_name];                    //获取管理员名称
$barcode=$_POST[barcode];                           //获取图书条形码
$bookName=$_POST[bookName];                         //获取图书名称
$typeid=$_POST[typeId];                             //获取图书类型 id 号
$author=$_POST[author];                             //获取图书作者
$translator=$_POST[translator];                     //获取图书译者
$isbn=$_POST[isbn];                                 //获取出版社 ISBN
$price=$_POST[price];                               //获取图书单价
$page=$_POST[page];                                 //获取图书页码
$bookcase=$_POST[bookcaseid];                       //获取图书书架 id 号
$inTime=date("Y-m-d");                              //设置图书更新日期为当前日期
$query=mysql_query("update tb_bookinfo set barcode='$barcode', bookName='$bookName' , typeid='$typeid',
author='$author', translator='$translator', ISBN='$isbn' , price='$price' , page='$page' , bookcase='$bookcaseid',
inTime='$inTime', operator='$operator' where id=$bid");            //更新数据表
echo "<script language='javascript'>alert('图书信息修改成功!');history.back();</script>";
?>
```

修改的图书信息后面的“修改”超链接，进入到修改图书信息页面。修改图书信息页面的运行结果如图 28.26 所示。

图 28.26　修改图书信息页面的运行结果

在查看图书信息列表页面中，添加“修改”超链接的代码如下：

（代码位置：光盘\TM\Instance\28\book_ok.php）

```
<a href="book_Modify.php?id=<?php echo $result[bookid];?>">修改</a>
```

在修改图书信息页面中修改图书信息后，单击“保存”按钮，提交表单信息到数据处理页 book_Modify_ok.php，应用 UPDATE 语句将修改的图书信息保存到数据表 tb_bookinfo 中，修改成功后，则弹出“图书信息修改成功！”提示信息，并将页面重定向到修改图书信息页。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\28\book_Modify_ok.php）

```
<?php
session_start();                                        //初始化 session 变量
include("conn/conn.php");                               //连接数据库文件
$bid=$_POST[bid];                                       //获取图书 id 号
$operator=$_SESSION[admin_name];                        //获取管理员名称
$barcode=$_POST[barcode];                               //获取图书条形码
$bookName=$_POST[bookName];                             //获取图书名称
$typeid=$_POST[typeId];                                 //获取图书类型 id 号
$author=$_POST[author];                                 //获取图书作者
$translator=$_POST[translator];                         //获取图书译者
$isbn=$_POST[isbn];                                     //获取出版社 ISBN
$price=$_POST[price];                                   //获取图书单价
$page=$_POST[page];                                     //获取图书页码
$bookcase=$_POST[bookcaseid];                           //获取图书书架 id 号
$inTime=date("Y-m-d");                                  //设置图书更新日期为当前日期
$query=mysql_query("update tb_bookinfo set barcode='$barcode', bookName='$bookName' , typeid='$typeid',
author='$author', translator='$translator', ISBN='$isbn' , price='$price' , page='$page' , bookcase='$bookcaseid',
inTime='$inTime', operator='$operator' where id=$bid");  //更新数据表
echo "<script language='javascript'>alert('图书信息修改成功!');history.back();</script>";
?>
```

28.7.6 删除图书信息的实现过程

在查看图书信息列表页面中，设置“删除”超链接的代码如下：

（代码位置：光盘\TM\Instance\28\book.php）

```
<a href="book_del.php?id=<?php echo $result[bookid];?>">删除</a>
```

单击想要删除的图书信息后面的“删除”超链接，提交表单信息到数据处理页 book_del.php，应用 DELETE 语句将指定的图书信息从数据表 tb_bookinfo 中删除，如果删除操作执行成功，则弹出“图书信息删除成功！”提示信息，并将页面重定向到图书信息列表页面。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\28\book_del.php）

```
<?php
include("conn/conn.php");                                              //连接数据库文件
$info_del=mysql_query("delete from tb_bookinfo where id=$_GET[id]");    //删除指定的图书信息
if($info_del){                                                         //如果信息删除成功，则弹出提示
    echo "<script language='javascript'>alert('图书信息删除成功!');history.back();</script> ";
}
?>
```

28.8 图书借还模块设计

视频讲解：光盘\TM\Video\第 28 章\图书借还模块设计.exe

28.8.1 图书借还模块概述

图书借还模块主要包括图书借阅、图书续借、图书归还、图书档案查询、图书借阅查询、借阅到期提醒 6 个功能。在图书借阅模块中的用户只有一种身份，即操作员，通过该身份可以进行图书借还等相关操作。图书借还模块的结构图如图 28.27 所示。

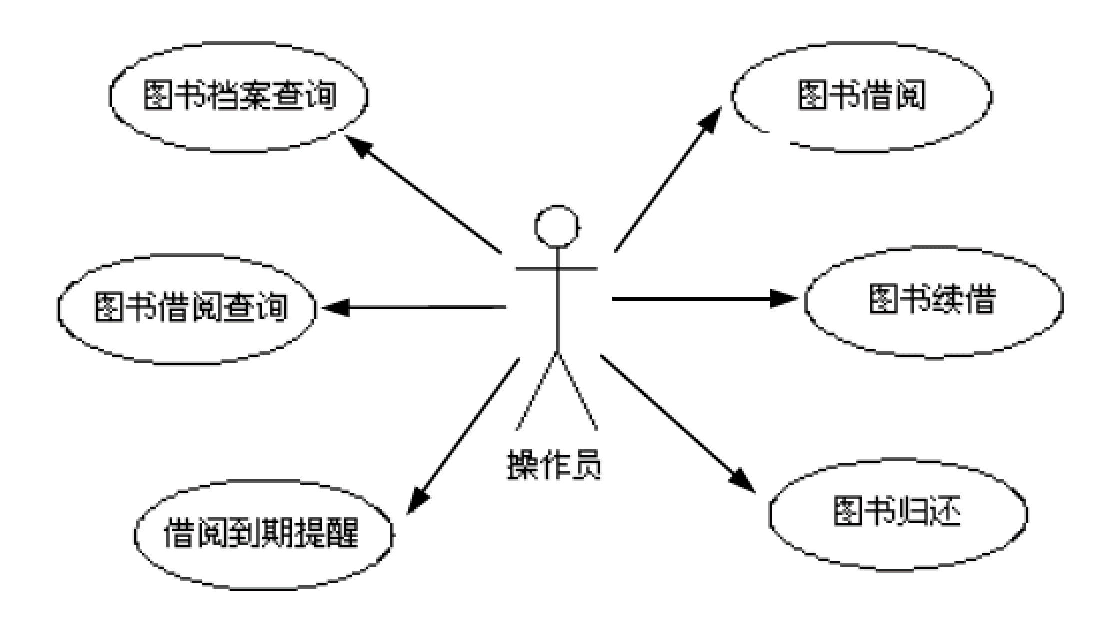
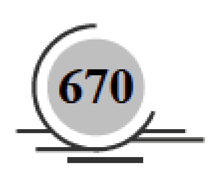


图 28.27 图书借还模块的结构图

28.8.2 图书借还模块中的多表查询技术

在图书借还模块中涉及到的数据表是 tb_borrow（图书借阅信息表）、tb_bookinfo（图书信息表）和 tb_reader（读者信息表），这 3 个数据表间通过相应的字段进行关联，如图 28.28 所示。



28.7.6 删除图书信息的实现过程

在查看图书信息列表页面中，设置“删除”超链接的代码如下：

（代码位置：光盘\TM\Instance\28\book.php）

```
<a href="book_del.php?id=<?php echo $result[bookid];?>">删除</a>
```

单击想要删除的图书信息后面的“删除”超链接，提交表单信息到数据处理页 book_del.php，应用 DELETE 语句将指定的图书信息从数据表 tb_bookinfo 中删除，如果删除操作执行成功，则弹出“图书信息删除成功！”提示信息，并将页面重定向到图书信息列表页面。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\28\book_del.php）

```
<?php
include("conn/conn.php");                                                    //连接数据库文件
$info_del=mysql_query("delete from tb_bookinfo where id=$_GET[id]");         //删除指定的图书信息
if($info_del){                                                               //如果信息删除成功，则弹出提示
    echo "<script language='javascript'>alert('图书信息删除成功!');history.back();</script> ";
}
?>
```

28.8 图书借还模块设计

视频讲解：光盘\TM\Video\第 28 章\图书借还模块设计.exe

28.8.1 图书借还模块概述

图书借还模块主要包括图书借阅、图书续借、图书归还、图书档案查询、图书借阅查询、借阅到期提醒 6 个功能。在图书借阅模块中的用户只有一种身份，即操作员，通过该身份可以进行图书借还等相关操作。图书借还模块的结构图如图 28.27 所示。

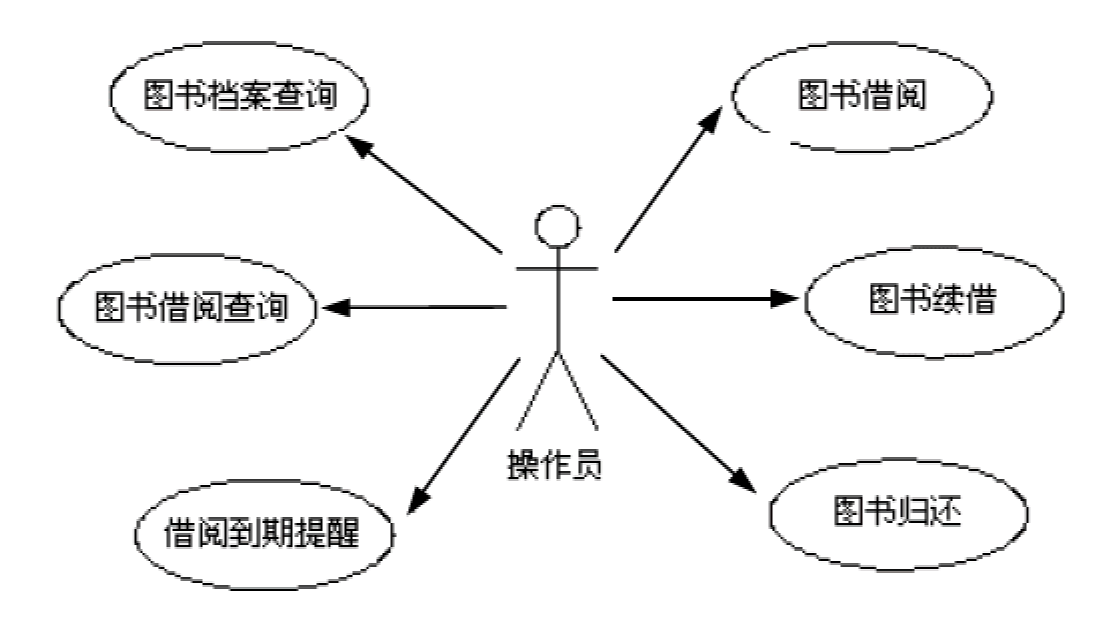
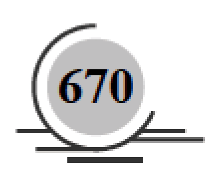


图 28.27 图书借还模块的结构图

28.8.2 图书借还模块中的多表查询技术

在图书借还模块中涉及到的数据表是 tb_borrow（图书借阅信息表）、tb_bookinfo（图书信息表）和 tb_reader（读者信息表），这 3 个数据表间通过相应的字段进行关联，如图 28.28 所示。



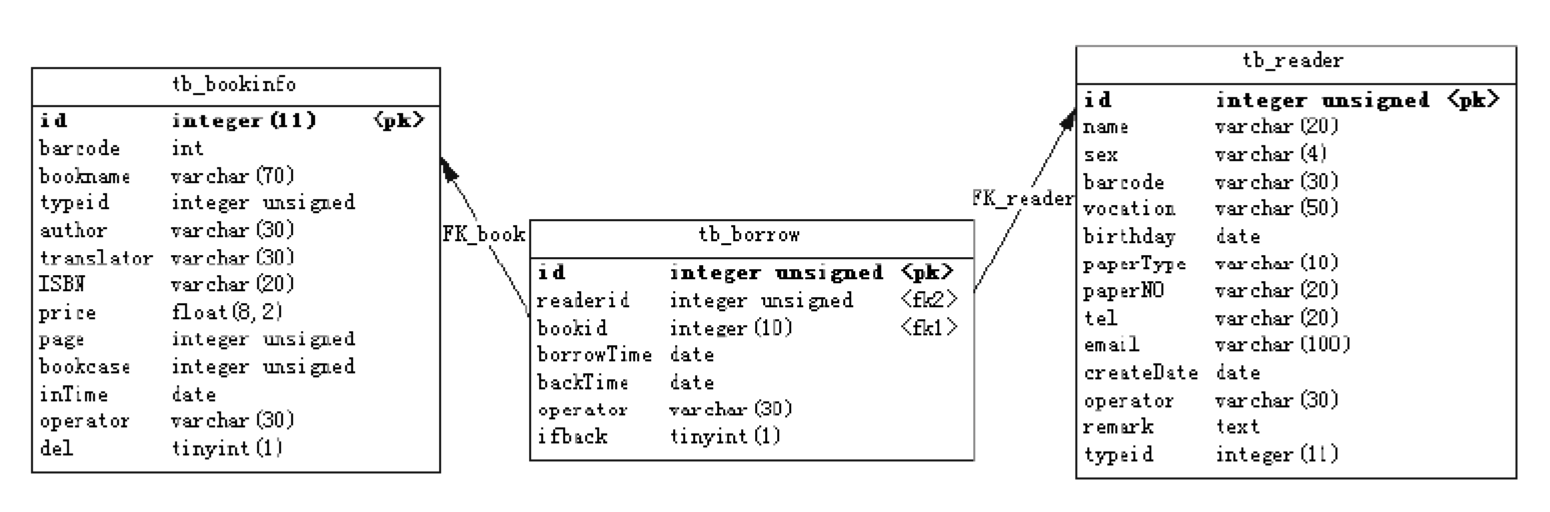


图 28.28　图书借还模块各表间关系图

28.8.3　图书借阅页面的实现过程

管理员登录后，选择“图书借还”/“图书借阅”命令，进入到图书借阅页面。在该页面中的“读者条形码”文本框中输入读者的条形码（如 1234561789）后，单击“确定”按钮，系统会自动检索出该读者的基本信息和未归还的借阅图书信息。如果检索到对应的读者信息，将其显示在页面中，此时输入图书的条形码或图书名称后，单击“确定”按钮，借阅指定的图书，运行结果如图 28.29 所示。



图 28.29　图书借阅页面的运行结果

说明　当读者借阅图书完毕后，单击“完成借阅”按钮，将重新载入图书借阅页面，当前页处于空信息状态，从而方便操作员进行下一个读者借阅图书的操作。

图书借阅页面可以分为两部分：一部分用于查询并显示读者信息；另一部分用于显示和添加读者的借阅信息。图书借阅页面在 Dreamweaver 中的设计效果如图 28.30 所示。



图 28.30　图书借阅页面的设计效果

在进行图书借阅时，系统要求每个读者只能同时借阅一定数量的图书，并且该数量由读者类型表 tb_readerType 中的可借数量 number 决定，所以笔者编写了自定义的 checkbook()函数，用于判断当前选择的读者是否还可以借阅新的图书，同时该函数还具有判断读者条形码或图书名称文本框是否为空的功能。其代码如下：

（代码位置：光盘\TM\Instance\28\book_del.php）

```
<script language="javascript">
function checkbook(form){                                      //自定义一个 JavaScript 函数 checkbook()
        if(form.barcode.value==""){                            //如果读者条形码为空
                alert("请输入读者条形码!");form.barcode.focus();return;   //弹出提示，焦点返回到条形码文本框
        }
        if(form.inputkey.value==""){                           //如果图书查询文本框的值为空
                alert("请输入查询关键字!");form.inputkey.focus();return; //弹出提示，焦点返回到图书查询文本框
        }
        if(form.number.value-form.borrowNumber.value<=0){      //如果图书的借阅数量超过了可借数量
                alert("您不能再借阅其他图书了!");return;        //弹出提示信息
        }
     form.submit();                                            //提交表单
  }
</script>
```

技巧 在 JavaScript 中比较两个数值型文本框的值时，不使用运算符 "=="，而是将这两个值相减，再判断其结果。

检索读者的基本信息和未归还的借阅图书信息的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\bookborrow.php）

```
$sql=mysql_query("select r.*,t.name as typename,t.number from tb_reader r left join tb_readerType t on r.typeid=t.id where r.barcode='$barcode'");
$info=mysql_fetch_array($sql);                          //检索读者信息
```

获取读者借阅信息的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\bookborrow.php）

```
$sql1=mysql_query("select r.*,borr.borrowTime,borr.backTime,book.bookname,book.price,pub.pubname,bc.name as bookcase from tb_borrow as borr join tb_bookinfo as book on book.id=borr.bookid join tb_publishing as pub on book.ISBN=pub.ISBN join tb_bookcase as bc on book.bookcase=bc.id join tb_reader as r on borr.readerid=r.id  where borr.readerid='$readerid' and borr.ifback=0");
$info1=mysql_fetch_array($sql1);                        //检索读者的借阅信息
$borrowNumber=mysql_num_rows($sql1);                    //获取结果集中行的数目
```

在"图书条形码"或"图书名称"文本框中输入图书条形码或图书名称后，单击"确定"按钮，检索图书信息是否存在，如果不存在，则向图书借阅信息表中添加该读者的图书借阅记录，完成图书的借阅操作；否则，弹出该图书不能被同一读者重复借阅的提示信息。图书借阅的具体代码如下：

（代码位置：光盘\TM\Instance\28\bookBorrow.php）

```
 <?php
if($_POST[inputkey]!=""){                               //如果"图书条形码"/"图书名称"后的文本框不为空
$f=$_POST[f];                                           //获取用户选择的条件值
❶    $inputkey=trim($_POST[inputkey]);                  //获取用户输入的查询关键字
$barcode=$_POST[barcode];                               //获取读者的条形码
$readerid=$_POST[readerid];                             //获取读者 id 号
❷    $borrowTime=date('Y-m-d');                         //图书的借阅时间为系统当前时间
❸    $backTime=date("Y-m-d",(time()+3600*24*30));       //归还图书日期为当前日期+30 天期限
$query=mysql_query("select * from tb_bookinfo where $f='$inputkey'");
$result=mysql_fetch_array($query);                      //检索图书信息是否存在
if($result==false){                                     //如果读者借阅的图书不存在，那么弹出提示信息
     echo "<script language='javascript'>alert('该图书不存在！');window.location.href='bookBorrow.php? barcode=$barcode'; </script>";
}
else{                                                   //检索该读者所借阅的图书是否与再借图书重复
     $query1=mysql_query("select  r.*,borr.borrowTime,borr.backTime,book.bookname,book.price,pub.pubname,
```

```
<script language="javascript">
function checkbook(form){                                            //自定义一个 JavaScript 函数 checkbook()
        if(form.barcode.value==""){                                  //如果读者条形码为空
                alert("请输入读者条形码!");form.barcode.focus();return;   //弹出提示，焦点返回到条形码文本框
        }
        if(form.inputkey.value==""){                                 //如果图书查询文本框的值为空
                alert("请输入查询关键字!");form.inputkey.focus();return; //弹出提示，焦点返回到图书查询文本框
        }
        if(form.number.value-form.borrowNumber.value<=0){            //如果图书的借阅数量超过了可借数量
                alert("您不能再借阅其他图书了!");return;                 //弹出提示信息
        }
    form.submit();                                                   //提交表单
  }
</script>
```

技巧 在 JavaScript 中比较两个数值型文本框的值时，不使用运算符“==”，而是将这两个值相减，再判断其结果。

检索读者的基本信息和未归还的借阅图书信息的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\bookborrow.php）

```
$sql=mysql_query("select r.*,t.name as typename,t.number from tb_reader r left join tb_readerType t on
r.typeid=t.id where r.barcode='$barcode'");
$info=mysql_fetch_array($sql);                                       //检索读者信息
```

获取读者借阅信息的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\bookborrow.php）

```
$sql1=mysql_query("select r.*,borr.borrowTime,borr.backTime,book.bookname,book.price,pub.pubname,bc.name
as bookcase from tb_borrow as borr join tb_bookinfo as book on book.id=borr.bookid join tb_publishing as pub
on book.ISBN=pub.ISBN join tb_bookcase as bc on book.bookcase=bc.id join tb_reader as r on
borr.readerid=r.id   where borr.readerid='$readerid' and borr.ifback=0");
$info1=mysql_fetch_array($sql1);                                     //检索读者的借阅信息
$borrowNumber=mysql_num_rows($sql1);                                 //获取结果集中行的数目
```

在“图书条形码”或“图书名称”文本框中输入图书条形码或图书名称后，单击“确定”按钮，检索图书信息是否存在，如果不存在，则向图书借阅信息表中添加该读者的图书借阅记录，完成图书的借阅操作；否则，弹出该图书不能被同一读者重复借阅的提示信息。图书借阅的具体代码如下：

（代码位置：光盘\TM\Instance\28\bookBorrow.php）

```
 <?php
if($_POST[inputkey]!=""){                                            //如果“图书条形码”/“图书名称”后的文本框不为空
$f=$_POST[f];                                                        //获取用户选择的条件值
❶     $inputkey=trim($_POST[inputkey]);                              //获取用户输入的查询关键字
$barcode=$_POST[barcode];                                            //获取读者的条形码
$readerid=$_POST[readerid];                                          //获取读者 id 号
❷     $borrowTime=date('Y-m-d');                                     //图书的借阅时间为系统当前时间
❸     $backTime=date("Y-m-d",(time()+3600*24*30));                   //归还图书日期为当前日期+30 天期限
$query=mysql_query("select * from tb_bookinfo where $f='$inputkey'");
$result=mysql_fetch_array($query);                                   //检索图书信息是否存在
if($result==false){                                                  //如果读者借阅的图书不存在，那么弹出提示信息
      echo "<script language='javascript'>alert('该图书不存在！');window.location.href='bookBorrow.php? barcode=
$barcode'; </script>";
}
else{                                                                //检索该读者所借阅的图书是否与再借图书重复
      $query1=mysql_query("select  r.*,borr.borrowTime,borr.backTime,book.bookname,book.price,pub.pubname,
```

```
bc.name as bookcase from tb_borrow as borr join tb_reader as r on borr.readerid=r.id join tb_bookinfo as book
on book.id=borr.bookid join tb_publishing as pub on book.ISBN=pub.ISBN  join tb_bookcase as bc on
book.bookcase=bc.id  where borr.bookid=$result[id] and borr.readerid=$readerid and ifback=0");
    $result1=mysql_fetch_array($query1);
    if($result1==true){                    //如果所借图书已被该读者借阅，那么提示不能重复借阅
        echo "<script language='javascript'>alert('该图书已经借阅！');window.location.href='bookBorrow.php?
barcode=$barcode';</script>";
    }
    else{                                  //否则，完成图书借阅操作，并弹出借阅成功提示信息
        $bookid=$result[id];               //将读者 id 号赋给一变量
        mysql_query("insert into
tb_borrow(readerid,bookid,borrowTime,backTime,operator,ifback)values('$readerid','$bookid','$borrowTime','$b
ackTime','$_SESSION[admin_name]',0)");     //向借阅信息表中添加一条借阅信息
        echo "<script language='javascript'>alert('图书借阅操作成功！');window.location.href='bookBorrow.php?
barcode=$barcode';</script>";
    }
    }
}
?>
```

说明

❶ trim()：删除字符串中首尾的空格或者其他字符，以达到精确查询。

❷ date('Y-m-d')：获取系统的当前日期为图书借阅的日期，并格式化日期格式。

❸ date("Y-m-d",(time()+3600*24*30))：归还图书日期=当前日期+30 天期限。当前日期时间戳应用 time()函数获取，30 天期限的时间戳等于 3600 秒 × 24 小时 × 30 天，并通过 date()函数格式化为指定日期格式。

28.8.4　图书续借页面的实现过程

管理员登录后，选择“图书借还”/“图书续借”命令，进入到图书续借页面。在该页面中的“读者条形码”文本框中输入读者的条形码（如 1234561789）后，单击“确定”按钮，系统会自动检索出该读者的基本信息和未归还的借阅图书信息。如果检索到对应的读者信息，则将其显示在页面中，此时单击“续借”超链接，即可续借指定图书（即将该图书的归还时间加上该书的可借天数 30 天计算得出）。图书续借页面的运行结果如图 28.31 所示。



图 28.31　图书续借页面的运行结果

```
bc.name as bookcase from tb_borrow as borr join tb_reader as r on borr.readerid=r.id join tb_bookinfo as book
on book.id=borr.bookid join tb_publishing as pub on book.ISBN=pub.ISBN   join tb_bookcase as bc on
book.bookcase=bc.id   where borr.bookid=$result[id] and borr.readerid=$readerid and ifback=0");
        $result1=mysql_fetch_array($query1);
        if($result1==true){                           //如果所借图书已被该读者借阅，那么提示不能重复借阅
                echo "<script language='javascript'>alert('该图书已经借阅！');window.location.href='bookBorrow.php?
barcode=$barcode';</script>";
        }
        else{                                         //否则，完成图书借阅操作，并弹出借阅成功提示信息
                $bookid=$result[id];                  //将读者 id 号赋给一变量
                mysql_query("insert into
tb_borrow(readerid,bookid,borrowTime,backTime,operator,ifback)values('$readerid','$bookid','$borrowTime','$b
ackTime','$_SESSION[admin_name]',0)");                //向借阅信息表中添加一条借阅信息
                echo "<script language='javascript'>alert('图书借阅操作成功！');window.location.href='bookBorrow.php?
barcode=$barcode';</script>";
        }
        }
}
?>
```

说明

❶ trim()：删除字符串中首尾的空格或者其他字符，以达到精确查询。

❷ date('Y-m-d')：获取系统的当前日期为图书借阅的日期，并格式化日期格式。

❸ date("Y-m-d",(time()+3600*24*30))：归还图书日期=当前日期+30 天期限。当前日期时间戳应用 time()函数获取，30 天期限的时间戳等于 3600 秒 × 24 小时 × 30 天，并通过 date()函数格式化为指定日期格式。

28.8.4　图书续借页面的实现过程

管理员登录后，选择“图书借还”/“图书续借”命令，进入到图书续借页面。在该页面中的“读者条形码”文本框中输入读者的条形码（如 1234561789）后，单击“确定”按钮，系统会自动检索出该读者的基本信息和未归还的借阅图书信息。如果检索到对应的读者信息，则将其显示在页面中，此时单击“续借”超链接，即可续借指定图书（即将该图书的归还时间加上该书的可借天数 30 天计算得出）。图书续借页面的运行结果如图 28.31 所示。



图 28.31　图书续借页面的运行结果

说明 当读者续借完图书后，单击“完成续借”按钮，将重新载入图书续借页面，当前页处于空信息状态，从而方便操作员进行下一个读者续借图书的操作。

图书续借页面的设计方法同图书借阅页面类似，不同的是，在图书续借页面中没有添加借阅图书的功能，而是添加了“续借”超链接。图书续借页面在 Dreamweaver 中的设计效果如图 28.32 所示。



图 28.32　图书续借页面的设计效果

单击“续借”超链接时，还需要将读者条形码、借阅 id 号和图书归还时间一同传递到图书续借的处理页 borrow_oncemore.php 中。其代码如下：

（代码位置：光盘\TM\Instance\28\bookRenew.php）

```
<a href="borrow_oncemore.php?barcode=<?php echo $info[barcode];?>&borrid=<?php echo
$info[borrid];?>&backTime=<?php echo $info[backTime];?>">续借</a>
```

检索读者信息和读者借阅信息的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\bookRenew.php）

```
$sql=mysql_query("select borr.id as borrid,borr.borrowTime,borr.backTime,borr.ifback,r.*,t.name as
typename,t.number,book.bookname,book.price,pub.pubname,bc.name as bookcase from tb_borrow as borr
join tb_reader r on borr.readerid=r.id join tb_readerType t on r.typeid=t.id join tb_bookinfo as book on
book.id=borr.bookid join tb_publishing as pub on book.ISBN=pub.ISBN join tb_bookcase as bc on
book.bookcase=bc.id where r.barcode='$barcode' and borr.ifback=0");
$info=mysql_fetch_array($sql);                                          //检索读者信息和借阅信息
```

单击“续借”超链接，提交到数据处理页 borrow_oncemore.php，用于完成图书的续借功能，主要通过更改图书的归还日期（即将该图书的归还时间加上该书的可借天数 30 天计算得出）实现。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\28\borrow_oncemore.php）

```
<?php
session_start();                                          //初始化 session 变量
include("conn/conn.php");                                 //连接数据库文件
$barcode=$_GET[barcode];                                  //获取图书条形码
$new=$_GET[backTime];                                     //获取图书归还时间
//更新续借期，将动态获取的还书日期转化为时间戳，然后再求出续借后的还书日期
$newbackTime=date("Y-m-d",(mktime(0, 0, 0, substr($new,5,2), substr($new,8,2), substr($new,0,4))+3600*24*30));
$borrid=$_GET[borrid];                                    //获取续借图书的 id 号
mysql_query("update tb_borrow set backTime='$newbackTime',ifback=0,operator='$_SESSION[admin_name]'
where id=$borrid");
echo "<script language='javascript'>alert('图书续借操作成功！');window.location.href='bookRenew.php?
barcode=$barcode';</script>";                             //弹出图书续借成功的提示信息
?>
```

28.8.5　图书归还页面的实现过程

管理员登录后，选择“图书借还”/“图书归还”命令，进入到图书归还页面。在该页面中的“读者条形码”文本框中输入读者的条形码（如 1234561789）后，单击“确定”按钮，系统会自动检索出该读者的基本信息和未归还的借阅图书信息。如果检索到对应的读者信息，则将其输出到浏览器，此时单击“归还”超链接，即可将指定图书归还。图书归还页面的运行结果如图 28.33 所示。



图 28.33　图书归还页面的运行结果

图书归还页面的设计方法同图书续借页面类似，不同的是，将图书续借页面中的“续借”超链接转化为“归还”超链接。在单击“归还”超链接时，也需要将读者条形码和借阅 id 号一同传递到图书归还处理页。其代码如下：

（代码位置：光盘\TM\Instance\28\bookBack.php）

```
<a href="bookBack_ok.php?borrid=<?php echo $info[borrid];?>&barcode=<?php echo $info[barcode];?>">归还</a>
```

检索读者信息及读者借阅信息的 SQL 语句如下：

（代码位置：光盘\TM\Instance\28\bookBack.php）

```
$sql=mysql_query("select borr.id as borrid,borr.borrowTime,borr.backTime,borr.ifback,r.*,t.name as typename,
t.number,book.bookname,book.price,pub.pubname,bc.name as bookcase from tb_borrow as borr join tb_reader
r on borr.readerid=r.id join tb_readerType t on r.typeid=t.id join tb_bookinfo as book on book.id=borr.bookid join
tb_publishing as
 pub on book.ISBN=pub.ISBN  join tb_bookcase as bc on book.bookcase=bc.id where r.barcode='$barcode'
and borr.ifback=0");
$info=mysql_fetch_array($sql);                                    //检索读者信息及该读者的借阅信息
```

单击“归还”超链接，即可将指定图书归还。数据处理页的代码如下：

（代码位置：光盘\TM\Instance\28\bookBack_ok.php）

```
<?php
session_start();                                                  //初始化 session 变量
include("conn/conn.php");                                         //连接数据库文件
$backTime=date("Y-m-d");                                          //归还图书日期
$borrid=$_GET[borrid];                                            //获取读者的 id 号
mysql_query("update  tb_borrow  set  backTime='$backTime',ifback=1,operator='$_SESSION[admin_name]'
where id=$borrid");                                               //更新读者的借阅信息
echo "<script language='javascript'>alert('图书归还操作成功！
');window.location.href='bookBack.php?barcode=$barcode';</script>";    ///弹出图书归还成功的提示信息
?>
```

28.8.6　图书借阅查询页面的实现过程

管理员登录后，选择“系统查询”/“图书借阅查询”命令，进入到图书借阅查询页面。图书借阅查询页面的运行结果如图 28.34 所示。在该页面中可以按指定的字段或某一时间段进行查询，同时还可以实现按指定字段及时间段进行综合条件查询。



图 28.34　图书借阅查询页面的运行结果

图书借阅查询页面主要用于收集查询条件和显示查询结果，并通过自定义的 JavaScript 函数验证输入的查询条件是否合法。该页面所涉及到的表单元素如表 28.5 所示。

表 28.5　图书借阅查询页面所涉及的表单元素

名　称	元 素 类 型	重 要 属 性	含　义
myform	form	method="post" action=""	表单
flag1	checkbox	value="a"	请选择查询依据
flag2	checkbox	value="b"	借阅时间
f	select	<option value="k.barcode" >图书条形码</option> <option value="k.bookname">图书名称</option> <option value="r.barcode">读者条形码</option> <option value="r.name">读者名称</option>	查询字段
key	text	size="50"	关键字
sdate	text	id="sdate"	开始日期
edate	text	id="edate"	结束日期
Submit	submit	onClick="return check(myform);"	“查询”按钮

在图书借阅查询页面中，指定查询条件后，提交表单信息到当前页。首先获取表单元素复选框 flag 的值，然后根据 flag 的值组合查询字符串。如果 flag1 的值等于 a，那么按指定的字段检索图书借阅信息；如果 flag2 的值等于 b，那么按指定的时间段检索图书借阅信息；如果 flag1 的值等于 a，并且 flag2 的值等于 b，那么按以上两个条件的综合条件检索图书借阅信息，并将查询结果输出到浏览器。其具体代码如下：

（代码位置：光盘\TM\Instance\28\borrowQuery.php）



28.8.6　图书借阅查询页面的实现过程

管理员登录后，选择“系统查询”/“图书借阅查询”命令，进入到图书借阅查询页面。图书借阅查询页面的运行结果如图 28.34 所示。在该页面中可以按指定的字段或某一时间段进行查询，同时还可以实现按指定字段及时间段进行综合条件查询。



图 28.34　图书借阅查询页面的运行结果

图书借阅查询页面主要用于收集查询条件和显示查询结果，并通过自定义的 JavaScript 函数验证输入的查询条件是否合法。该页面所涉及到的表单元素如表 28.5 所示。

表 28.5　图书借阅查询页面所涉及的表单元素

名　　称	元 素 类 型	重 要 属 性	含　　义
myform	form	method="post" action=""	表单
flag1	checkbox	value="a"	请选择查询依据
flag2	checkbox	value="b"	借阅时间
f	select	<option value="k.barcode" >图书条形码</option> <option value="k.bookname">图书名称</option> <option value="r.barcode">读者条形码</option> <option value="r.name">读者名称</option>	查询字段
key	text	size="50"	关键字
sdate	text	id="sdate"	开始日期
edate	text	id="edate"	结束日期
Submit	submit	onClick="return check(myform);"	“查询”按钮

在图书借阅查询页面中，指定查询条件后，提交表单信息到当前页。首先获取表单元素复选框 flag 的值，然后根据 flag 的值组合查询字符串。如果 flag1 的值等于 a，那么按指定的字段检索图书借阅信息；如果 flag2 的值等于 b，那么按指定的时间段检索图书借阅信息；如果 flag1 的值等于 a，并且 flag2 的值等于 b，那么按以上两个条件的综合条件检索图书借阅信息，并将查询结果输出到浏览器。其具体代码如下：

（代码位置：光盘\TM\Instance\28\borrowQuery.php）



```
    <?php
    include("conn/conn.php");                                    //连接数据库文件
    $sql=mysql_query("select  b.borrowTime,b.backTime,b.ifback,r.barcode  as  readerbarcode,r.name,k.id,
k.barcode,k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id");
                                                                 //查询图书借阅信息
if($_POST[Submit]!=""){                                          //如果提交了表单，则执行以下操作
     $f=$_POST[f];                                               //获取操作员选择的查询条件
     $key1=$_POST[key1];                                         //获取查询关键字
     $sdate=$_POST[sdate];                                       //获取借阅的起始日期
     $edate=$_POST[edate];                                       //获取借阅的结束日期
     $flag1=$_POST[flag1];                                       //获取按指定条件查询的复选框值
     $flag2=$_POST[flag2];                                       //获取按日期查询的复选框值
     if($flag1=="a"){                                            //如果按指定条件查询，则执行以下语句
     $sql=mysql_query("select b.borrowTime,b.backTime,b.ifback,r.barcode as readerbarcode,r.name,k.id,k.barcode,
k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id where $f
like '%$key1%'");
     }
     if($flag2=="b"){                                            //如果按时间段查询，则执行以下语句
     $sql=mysql_query("select  b.borrowTime,b.backTime,b.ifback,r.barcode  as  readerbarcode,r.name,k.id,
k.barcode,k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id
where borrowTime between '$sdate' and '$edate'");
     }
     if($flag1=="a" && $flag2=="b"){                             //如果按综合条件查询，则执行以下语句
     $sql=mysql_query("select  b.borrowTime,b.backTime,b.ifback,r.barcode  as  readerbarcode,r.name,k.id,
k.barcode,k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id
where borrowTime between '$sdate' and '$edate' and $f like '%$key1%'");
     }
}
$result=mysql_fetch_array($sql);                                 //检索查询结果
if($result==false){                                              //如果查询结果不存在，则弹出提示信息
?>
<table width="100%" height="30"  border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td height="36" align="center">暂无图书借阅信息！</td>
  </tr>
</table>
<?php
}else{                                                           //否则，输出图书借阅信息
?>
<table  width="1723"  border="1"  cellpadding="0"  cellspacing="0"  bordercolor="#FFFFFF"  bordercolordark=
"#D2E3E6" bordercolorlight="#FFFFFF">
  <tr align="center" bgcolor="#D0E9F8">
    <td width="13%">图书条形码</td>
    <td width="217%">图书名称</td>
    <td width="15%">读者条形码</td>
    <td width="11%">读者名称</td>
    <td width="13%">借阅时间</td>
    <td width="11%">归还时间</td>
    <td width="10%">是否归还</td>
  </tr>
<?php
do{
if($result[ifback]=="0"){                                        //如果“是否归还”等于 0，则输出“未归还”
     $ifbackstr="未归还";
```

```
    <?php
    include("conn/conn.php");                                          //连接数据库文件
    $sql=mysql_query("select  b.borrowTime,b.backTime,b.ifback,r.barcode  as  readerbarcode,r.name,k.id,
k.barcode,k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id");
                                                                       //查询图书借阅信息
if($_POST[Submit]!=""){                                                //如果提交了表单，则执行以下操作
     $f=$_POST[f];                                                     //获取操作员选择的查询条件
     $key1=$_POST[key1];                                               //获取查询关键字
     $sdate=$_POST[sdate];                                             //获取借阅的起始日期
     $edate=$_POST[edate];                                             //获取借阅的结束日期
     $flag1=$_POST[flag1];                                             //获取按指定条件查询的复选框值
     $flag2=$_POST[flag2];                                             //获取按日期查询的复选框值
     if($flag1=="a"){                                                  //如果按指定条件查询，则执行以下语句
     $sql=mysql_query("select b.borrowTime,b.backTime,b.ifback,r.barcode as readerbarcode,r.name,k.id,k.barcode,
k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id where $f
like '%$key1%'");
     }
     if($flag2=="b"){                                                  //如果按时间段查询，则执行以下语句
     $sql=mysql_query("select  b.borrowTime,b.backTime,b.ifback,r.barcode  as  readerbarcode,r.name,k.id,
k.barcode,k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id
where borrowTime between '$sdate' and '$edate'");
     }
     if($flag1=="a" && $flag2=="b"){                                   //如果按综合条件查询，则执行以下语句
     $sql=mysql_query("select  b.borrowTime,b.backTime,b.ifback,r.barcode  as  readerbarcode,r.name,k.id,
k.barcode,k.bookname from tb_borrow b join tb_reader r on b.readerid=r.id join tb_bookinfo k on b.bookid=k.id
where borrowTime between '$sdate' and '$edate' and $f like '%$key1%'");
     }
}
$result=mysql_fetch_array($sql);                                       //检索查询结果
if($result==false){                                                    //如果查询结果不存在，则弹出提示信息
?>
<table width="100%" height="30"  border="0" cellpadding="0" cellspacing="0">
  <tr>
    <td height="36" align="center">暂无图书借阅信息！</td>
  </tr>
</table>
<?php
}else{                                                                 //否则，输出图书借阅信息
?>
<table  width="1723"  border="1"  cellpadding="0"  cellspacing="0"  bordercolor="#FFFFFF"  bordercolordark=
"#D2E3E6" bordercolorlight="#FFFFFF">
  <tr align="center" bgcolor="#D0E9F8">
    <td width="13%">图书条形码</td>
    <td width="217%">图书名称</td>
    <td width="15%">读者条形码</td>
    <td width="11%">读者名称</td>
    <td width="13%">借阅时间</td>
    <td width="11%">归还时间</td>
    <td width="10%">是否归还</td>
  </tr>
<?php
do{
if($result[ifback]=="0"){                                              //如果“是否归还”等于 0，则输出“未归还”
     $ifbackstr="未归还";
```

```
}else{                                                //如果“是否归还”等于 1，则输出“已归还”
        $ifbackstr="已归还";
}
?>
/* *******************************输出符合查询条件的记录**************************** */
<tr>
    <td style="padding:5px;"> <?php echo $result[barcode];?></td>
    <td style="padding:5px;"><a href="book_look.php?id=<?php echo $result[id]; ?>"><?php echo
$result[bookname];?></a></td>
    <td style="padding:5px;"> <?php echo $result[readerbarcode];?></td>
    <td style="padding:5px;"> <?php echo $result[name];?></td>
    <td style="padding:5px;"> <?php echo $result[borrowTime];?></td>
    <td style="padding:5px;"> <?php echo $result[backTime];?></td>
    <td style="padding:5px;"> <?php echo $ifbackstr;?></td>
</tr>
/* ******************************************************************************** */
<?php
        }while($result=mysql_fetch_array($sql));
}
?>
```

28.9 小　　结

本章运用软件工程的设计思想，通过一个完整的图书馆管理系统引导读者深入了解系统的开发流程。在该系统的实现过程中，除了应用一些基本的 PHP 技术外，还涉及一些独特的技术细节，如权限设置、多表查询技术等，希望读者掌握并能在实际的操作中灵活应用，举一反三。